Requirements Eng
DOI 10.1007/s00766-015-0220-8

ORIGINAL ARTICLE

Reusable knowledge in security requirements engineering: a systematic mapping study

Amina Souag · Raúl Mazo · Camille Salinesi · Isabelle Comyn-Wattiau

Received: 31 January 2014 / Accepted: 14 January 2015

Abstract Security is a concern that must be taken into consideration starting from the early stages of system development. Over the last two decades, researchers and engineers have developed a considerable number of methods for security requirements engineering. Some of them rely on the (re)use of security knowledge. Despite some existing surveys about security requirements engineering, there is not yet any reference for researchers and practitioners that presents in a systematic way the existing proposals, techniques, and tools related to security knowledge reuse in security requirements engineering. The aim of this paper is to fill this gap by looking into drawing a picture of the literature on knowledge and reuse in security requirements engineering. The questions we address are related to methods, techniques, modeling frameworks, and tools for and by reuse in security requirements engineering. We address these questions through a systematic mapping study. The mapping study was a literature review conducted with the goal of identifying, analyzing, and categorizing state-of-the-art research on our topic. This mapping study analyzes more than thirty approaches, covering 20 years of research in security requirements engineering. The contributions can be summarized as follows: (1) A framework was defined for analyzing and comparing the different proposals as well as categorizing future contributions related to knowledge reuse and security requirements engineering; (2) the different forms of knowledge representation and reuse were identified; and (3) previous surveys were updated. We conclude that most methods should introduce more reusable knowledge to manage security requirements.

Keywords Reusability · Security requirements · Knowledge · Ontologies · Patterns · Templates

1 Introduction and motivation

There is a clear trend nowadays to consider systems security at the requirements engineering stage. Formerly, security requirements were considered as technical choices made during implementation, resulting in late and expensive attempts to shoehorn security into the system in progress. This is particularly true with information systems (IS). Security requirements engineering (SRE) allows IS developers to predict the threats, their consequences and countermeasures before a system is in place, rather than as a reaction to possibly disastrous attacks [1]. SRE is concerned with protecting assets from harm [2]. To cope with these issues, an increasing number of publications, conference tracks, and workshops in recent years point out the growing interest of researchers and practitioners in providing SRE processes with various frameworks and methods. Some of them are extensions of goal-oriented approaches, like secure i* [3], secure tropos [4], KAOS, and anti-models [5]. Others are built on the object paradigm, mainly UML-based, such as misuse cases [6], security use cases [7], secure UML [8], and UMLSec [9].

At a high level of abstraction, every application tends to have the same basic kinds of *vulnerable assets* (e.g., data, communications, services, hardware components, and personnel). Similarly, these vulnerable assets tend to be subject to the same basic kinds of security *threats* (e.g.,

A. Souag (✉) · R. Mazo · C. Salinesi
CRI, Panthéon Sorbonne University, Paris, France
e-mail: aminasouag@gmail.com

I. Comyn-Wattiau
CEDRIC-CNAM & ESSEC Business School, Paris, France

Published online: 25 February 2015

theft, vandalism, unauthorized disclosure, destruction, fraud, extortion, and espionage) from attacks by the same basic kinds of *attackers* (e.g., hackers, crackers, disgruntled employees, international cyber terrorists, and industrial spies) who can be profiled with motivations and their typical levels of expertise and tools. Furthermore, security requirements tend to be even more standardized than their associated mechanisms. For example, to address the identification and authentication requirements, one may have several choices of architectural mechanisms beyond user ID and passwords [10]. Based on these facts, reuse of requirements could lead to significant savings in development time and cost [11]. Nowadays, the research community in SRE as well as practitioners has a vague idea of existing literature for handling knowledge reuse among existing SRE approaches. For instance, a quick research indicates that some of the approaches propose a catalog of attacks [12], while others rely on patterns [13]. However, a systematic mapping study and analysis of existing security requirements engineering methods that make (re)use of knowledge is still lacking.

This paper presents a structured and systematic mapping study of several articles related to knowledge reuse and security requirements engineering from the last two decades.

The goals of this research can be summarized as follows: (1) to update existing literature surveys related to SRE with recent researches, (2) to identify the (re)used knowledge in SRE, (3) to distinguish the different types of knowledge reuse structures in SRE, and (4) to understand their use. Our research method considers the state of the art in security requirements literature. More specifically, this mapping study must find answers to the following questions: Do the security requirements engineering methods rely on reusability of knowledge? What are the reusable elements? How are they represented, modeled? How are they (re)used? Are the knowledge-based approaches tool supported?

A framework was defined to understand the different proposals and classify new contributions in the future. Over 100 proposals were analyzed from which the paper reports the knowledge reuse situation of 30 methods.

The main target audiences of this mapping study are researchers and especially Ph.D. students in the field of security requirements, but also tool developers or practitioners who are interested in the field.

The paper is structured as follows. Section 2 presents a short introduction to knowledge reuse in security requirements engineering. Section 3 gives an overview of our research method. Section 4 presents the process and results of the conducted systematic mapping study to get an overview of existing security requirements approaches based on knowledge reuse. Section 5 summarizes the results and answers of the research questions. Section 6 reports the related works. Section 7 discusses threats to validity of our mapping study. Finally, Sect. 8 concludes this paper.

2 Knowledge reuse in security requirements engineering

Back in 1993, the second International Workshop on Software Reusability was held in Lucca, Italy. Most of the papers presented at this event focused on reusing code, design, or architecture. In other words, the thinking was that mainly the hard artifacts—code, object, and so on—could be reused. Very few papers looked at the idea of reuse earlier in the IS life cycle, namely reusing requirements themselves. Active areas of reuse research in the past 20 years include reuse libraries, domain engineering methods and tools, reuse design, design patterns, domain-specific software architecture, software componentry [14], generators, measurement, and experimentation [15].

Nowadays, the practice of reuse is moving upstream, and reuse is also concerned with more abstract artifacts. Requirements are commonly recycled; patterns are exchanged on the Internet. The notion of reuse at the requirements stage is largely accepted by many within the community as a desirable aim [16]. For instance, a working conference on patterns (Pattern Languages of Programs) is held twice a year and results in the sharing of knowledge and publication of new patterns [17].

Requirement reuse can be defined as either taking requirements that have been written for previous projects and then using them for a new project, or writing requirements from scratch at a reasonable level of generality and abstraction in order to use them over different projects. For instance, it is possible to reuse different types of data, ranging from business requirements and functional requirements to use cases and test cases. Since requirements engineering is the first phase in the software development process, requirements reuse can empower the software life cycle. Previous research [18] has pointed out that reusing the first software products and processes implemented in a development project can have an impact on the life cycle from two basic points of view: (a) allowing the software development resources to be more profitable and (b) promoting reuse-based development across the entire software process.

During the last decade, given the common nature of security problems across applications and application domains, researchers paid some attention to the benefits of reuse in SRE process [10]. Security knowledge is hard to acquire. In addition to awareness about potential attacks, designing security-critical systems requires knowledge and

security expertise in various fields such as computer networks, operating systems, communication protocols, cryptography algorithms, and access control methods. Reuse combined with pre-defined structured knowledge can make the job of requirements engineers much easier and faster, since they usually lack security expertise and skills. However, one should be careful when structuring reusable knowledge—it has to be of a high quality. Otherwise, it might end up introducing new security problems. A clear distinction must be made between engineering "for" reuse and engineering "by" reuse [19].

Structuring security knowledge helps the knowledge consumer to browse the content and to find the relevant information more efficiently. Different knowledge representations exist in the literature. Patterns of recurring attacks and vulnerabilities have been identified by longtime software security practitioners [20]. Security templates of a high level of abstraction were also introduced for reuse purposes [10]. Various other approaches for managing security knowledge and reuse exist in the literature, such as taxonomies, ontologies, standards, and guidelines.

3 Research method

A fair amount of publications, conference tracks, and workshops in SRE appeared during the last decade, revealing a steady interest of both researchers and practitioners in that domain. Unfortunately, it remains difficult to have more than a vague idea about what is available in terms of reuse of security requirements and to position research with respect to available practices in order to choose appropriate practice. One difficulty is due to the fact that these issues are addressed by several communities: the requirements engineering community, the software engineering community, the information systems community, and the computer security community.

Our research method aimed at analyzing and identifying the available literature on security requirements research and categorizing it in a systematic way. The systematic mAPping study (SMAP) was conducted between August 2013 and December 2013. We applied the mapping studies guidelines proposed by Petersen et al. [21], which compares the methods used in mapping studies and systematic literature reviews. The SMAP reported in this paper was performed based on these guidelines (cf. Fig. 1), to identify questions and answers raised by the research community on knowledge (re)use in SRE.

Reviewing existing research in a fully objective way is not possible. A systematic study, such as the one outlined in Fig. 1, however, makes the process less subjective by using pre-defined data forms and criteria that narrow the scope for personal interpretation.

Mapping studies must be distinguished from systematic literature reviews in several ways. Systematic literature reviews (SLR) have been defined as "a means of identifying, evaluating, and interpreting all available research relevant to a particular research question, or topic area, or phenomenon of interest" [22]. Mapping studies are a special kind of SLR that use the same basic methodology as SLRs but aim to identify and classify all research related to a broad software engineering topic rather than answering questions about the relative merits of competing technologies as addressed by conventional SLRs [23, 24]. SMAPs are intended to provide an overview of a topic area and identify whether there are sub-topics with sufficient primary studies to conduct conventional SLRs and also to identify sub-topics where more primary studies are needed.

Overall, the main phases of our systematic mapping study were: defining research questions, conducting the search for relevant papers, screening papers, key wording of abstracts, extracting data, planning mapping, conducting, and reporting. Figure 1 presents the process structure of our SMAP.

A key element, in the guidelines proposed by Petersen et al. [21], is the definition of the *research questions* (*research scope*). Research questions should reflect and reply to the main goals of a SMAP in providing an overview of a research area, identify the quantity and type of research and results available within it. The search for *primary studies* (*all papers*) is conducted thanks to search strings on scientific databases or browsing manually through relevant conference proceedings or journal publications. We performed screening of papers for *inclusion and exclusion* (*relevant papers*). In this step, inclusion and exclusion criteria are used to exclude studies that are not relevant to answer the research questions. *Key wording* using abstracts is a way to reduce the time needed in developing the *classification scheme* and ensuring that the scheme takes the existing studies into account. The process ends up with

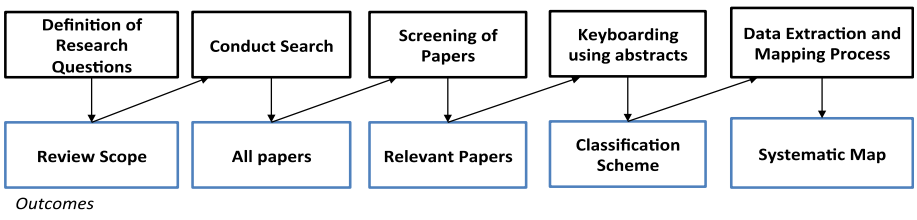

Fig. 1 Systematic mapping process carried out in this paper, applied from [21]

the *data extraction and mapping of studies;* here, the classification scheme evolves by adding new categories or mapping and splitting existing categories. More practical details on how we addressed these issues in our SMAP are given in the next section.

4 Reusable security knowledge: a systematic mapping study

The review includes publications reporting on existing approaches and tools as well as publications discussing research issues for security requirements and knowledge reuse in SRE. The SMAP was conducted in 24 relevant sources (the detailed list of the sources can be consulted in the cell "Digital library/resource" in Table 4 in the "Appendix"). The total retrieved number of publications is **158** using well-defined search criteria (which will be presented later). From these 158 publications, **95** papers were chosen for further analyses based on our set of selection criteria. The complete list of all 95 retrieved publications and details about the retrieved searches can be found in the "Appendix" (Table 4).

4.1 Conducting the systematic mapping study

The main goal for conducting the systematic mapping study was to get an extensive overview of existing knowledge-based approaches and tools for security requirements engineering and to understand key issues for security requirements elicitation and analysis considering the (re)use of knowledge in these practices. This systematic mapping study was developed using the following elements:

4.1.1 Definition of research questions

Our previous research led us to the conclusion that we need to reinforce security requirements engineering by enriching the process with specific knowledge on security. This knowledge is necessary to take into account security requirements early and consistently. It is generally not well known by information systems designers. Hence, we want to understand the current state of the art in this field. More specifically, we want to evaluate whether the security knowledge can be reused (RQ1). A deep analysis of this question requires that we elicit how this knowledge is represented (RQ2) and reused (RQ3). Moreover, can the whole knowledge be reused (RQ4)? Can it be reused automatically (RQ5)? Finally, what can be improved in current approaches (RQ6)? RQ2, RQ3, RQ4, RQ5, and RQ6 only make sense if RQ1 is answerable and if the answer is yes. RQ2, RQ3, RQ4, and RQ5 are necessary to understand which knowledge is currently reused, and RQ6 sketches avenues for our future research. Thus, our systematic review was guided by the following research questions, for each SRE method, and for SRE methods overall:

RQ1. Does the security requirements engineering method rely on reusable security knowledge? How many papers handled knowledge reuse in SRE? **(Knowledge reliance)**
RQ2. How is the reused knowledge represented? What are the proportions of each knowledge representation form? **(Form of representation)**
RQ3. What are the techniques for (re)using the knowledge and their proportion? **(Technique)**
RQ4. What are the main reused elements and their proportion? **(Reusable knowledge)**
RQ5. Is it tool supported? Are there many tools for SRE overall? **(Automation)**
RQ6. What are the new challenges regarding security knowledge (re)use in SRE?

Research question RQ1 checks, among the different existing proposals, whether the security engineering method at hand relies on the (re)use of knowledge. It also looks for the number of papers that rely on the (re)use of knowledge. RQ2 finds how (and how much) the (re)used knowledge is represented (modeling language, representation of requirements, etc.). RQ3 identifies how the security knowledge is (re)used. RQ4 reports what the main reusable elements found in proposals identified in RQ1 are: for instance, security requirements, threat models, or common vulnerabilities. RQ5 checks whether the SRE method offers automated support for the reuse of knowledge. It also examines the number of papers that propose tools for reuse in SRE. Finally, RQ6 extracts from the papers some new challenges that the SRE community should face in the future.

4.1.2 Search for primary studies

To *search for primary studies (all papers)*, the sources (presented in the "Appendix") were selected based on an analysis of security requirements literature. Our sources were extracted from digital libraries such as ACM Digital Library, Science Direct, IEEE Xplore, IEEE Computer Society, SpringerLink and DBLP; we chose those because our university had a subscription to them. Also journals, conferences, and workshops of the domain such as RE, REFSQ, ARES, and Requirements Engineering Journal were considered. These sources were chosen based on a pre-search on Google Scholar in addition to consulting the citations of existing SLRs and other SMAPs. Relevant books and reports were explored further. For all primary studies found in these sources, we also followed their

relevant cited references to find additional contributions outside the above-mentioned subset. All searches have been conducted on publications appeared between 2000 and 2013, thus covering over 13 years of SRE research.

Depending on the source, different search terms were used. For the more general conferences and for journals, we used the search terms "`reuse security requirements`," "`knowledge security requirements`," "`reusability in security requirements`," or "`knowledge reusability security requirements`" appearing in the full text of the publications (excluding references). In conferences and journals related to SRE, the search term was iteratively refined, for example leading to the search terms "`ontologies for security requirements`," "`pattern security requirements`," "`reuse misuse cases`," "`knowledge security use cases`," or "`reuse secure tropos`."

4.1.3 Screening of papers

Search for primary studies lead to 158 articles, many of which were irrelevant. Screening for papers based on the title and succinct review of the abstract, in addition to the reliance on our inclusion and exclusion criteria, reduced the number of relevant papers. The screening process was performed by two of the authors (team 1) and validated by the other two authors (team 2).

The following restrictions and quality criteria for including/excluding publications were defined:

- (*Restriction R1*) The study only includes papers available in electronic form. Books were analyzed based on information available online and using the hard copy versions.
- (*Restriction R2*) Only publications written in English were included.
- *(Quality criterion Q1)* Each publication was checked for completeness. Publications containing several unsupported claims or frequently referring to existing work without providing citations were excluded.
- (*Quality criterion Q2*) Articles related to the topic of this paper published between January 1, 2000, and August 31, 2013, were included: (1) papers proposing any method for SRE; (2) papers proposing knowledge reuse-based methods for SRE; (3) papers proposing automation of any (knowledge reuse-based) SRE.
- (*Quality criterion Q3*) Works of the same authors with very similar content were included and grouped under the same category (method).
- (*Quality criterion Q4*) Some articles were intentionally excluded to keep the level of the SMAP manageable, in particular when the proposition was not relevant enough to the topic of this paper.

Ninety-five searches in 24 sources were carried out using the search terms described above. In total, 158 publications were retrieved, out of which 21 were found not directly in the 24 sources but by following relevant cited references. Figure 2 shows the distribution of research results related to security requirements engineering and reuse between 2000 and 2013. The figure also shows that between 2004 and 2007, a great number of publications were published; thus, the well-known approaches for security requirements engineering appeared in this period. Table 4 (in the "Appendix") presents the retrieved and selected publications for each source.

4.1.4 Data classification

The retrieved publications were first analyzed regarding the restrictions R1–R2. The remaining publications were carefully assessed regarding quality criterion Q1. For each retrieved publication, the following standard information was collected in a data extraction form:

- Date of search, source, and used search term.
- Authors, title, and publication year.
- Type of publication (conference, workshop, journal, report, or book).
- Short summary (main claims, presented approach/tool).
- Restrictions R1, R2 (yes or no)?
- Quality criteria Q1, Q2, Q3, Q4 fulfilled (yes or no)?
- Addressed research question(s).
- Selected (yes or no)? Based on restrictions and quality criteria.
- Comments/rationale regarding selection.
- Need for tools. Does the publication stress the need for support (yes or no)?

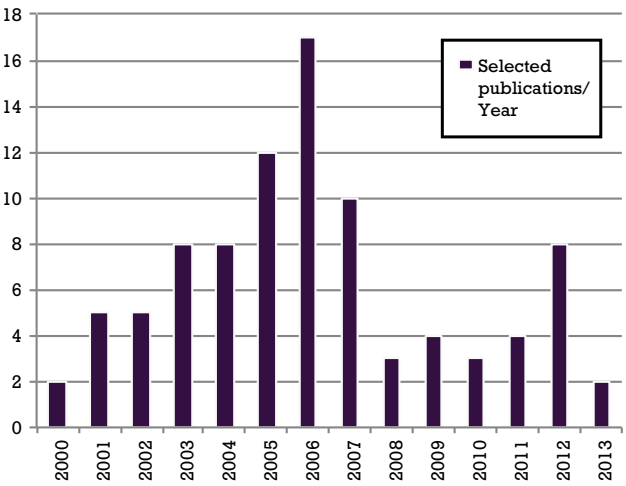

Fig. 2 Number of selected publications on (knowledge-based) security requirements engineering (2000–2013)

For each selected publication, the following additional information was captured in a second form to increase confidence regarding their relevance for security requirements engineering elicitation and analysis: The main focus of the publication is on security requirements (yes) versus security requirements are only addressed as part of a broader approach (no)?

Searching the security requirements approaches and (re)usable knowledge-based security requirements approaches conferences led to 158 papers, out of which 95 (60 %) were related specifically to security requirements approaches. Among these 95 papers, 29 papers (31 %) addressed reuse of knowledge for security requirements. Searching conferences led to the largest set of results: 39 papers (41 %) out of 70 papers found. Note that the selected conference papers were mainly found in two main conferences proceedings: The International Conference on Availability, Reliability and Security (ARES) with 13 papers found out of which 8 were selected; the International Conference on Requirements Engineering (RE) with 24 conference papers found and 7 selected.

The number of selected journal papers was 20 (21 %) out of 31 papers found. The total number of workshop papers found was 21, out of which 11 (12 %) were selected. Fifteen out of 18 (16 %) relevant technical reports were also considered in our search. Books and book chapters were taken in consideration too: out of 18 retrieved sources, 10 (10 %) were selected.

Table 4 in the "Appendix" gives all the details about the retrieved publications, their types and the ones selected. Figure 3 summarizes the statistical results of all selected papers by categories (books, conferences, workshops, and reports).

For the selected papers, we were also interested in the type of the research. As recommended by Petersen et al. [21], we adapted the classification system developed by Wieringa et al. [25] for requirements engineering paper classification. The papers were thereby classified into:

- Solution proposal: papers that discuss new or revised techniques,
- Philosophical: papers that sketch a new way of looking at things, a new conceptual framework, etc.
- Evaluation research: papers that investigate a practice or an implementation in practice and report the lessons learned.
- Validation research: papers that investigate the properties of a solution proposal that has not yet been deployed in practice.
- Opinion papers: papers that contain the author's opinion about a research or practice subject.
- Experience: papers that are often from industry practitioners or researchers who have used their tools in practice. They report how something was done in practice.

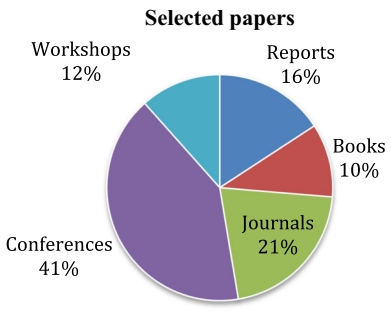

Fig. 3 Relative share of the various paper types in the selected set (Books, Conferences, Workshops, and Reports)

Some papers covered two categories. For example, a paper may be at the same time a "solution proposal" and "validation research." In such cases, we labeled them "solution proposal + validation." Conversely, some papers could not be linked to any category since they were exclusively presenting tools. Thus, we decided to use the label "tool."

Table 1 summarizes the results of the classification. Most of the papers are solution proposals (41 %), few of which are validated (22.1 %). Evaluation researches that investigated the practices in industry are only (10.5 %). Eight papers were exclusively presenting tools.

4.2 A framework for analyzing and comparing knowledge reuse in SRE (data extraction and mapping of studies)

Extracting the data, while surveying in depth the different approaches for SRE with regards to knowledge reuse, allows us to define the different categories covered by the study and construct the map (i.e., a framework for analyzing and comparing knowledge reuse in SRE). The framework shown in Fig. 4 is structured around facets that capture individual dimensions related to knowledge reuse in SRE. This framework makes it possible to organize the different methods, techniques, and tools for knowledge reuse in SRE around different axes that were identified through the SMAP and appeared relevant to us.

4.2.1 Knowledge

The knowledge facet identifies the different knowledge (re)used in SRE. This facet was organized under three main sub-dimensions (by analogy to the classification framework proposed by Dubois et al. [26]):

- Organization & Assets: All the knowledge related to the organization, its assets, its actors can be (re)used over different projects.

Table 1 Type and number of selected papers

	Solution Proposal	Philosophical	Evaluation	Validation	Experience	Opinion	Tool	Solution + validation + tool	Solution + validation
Book (chapter)	5	2	1	0	0	0	2	1	0
Journal	10	0	2	0	1	0	0	2	5
Conference	15	2	4	0	1	0	5	0	10
Workshop	4	2	1	2	0	0	0	2	1
Report	5	2	2	0	0	0	1	0	5
Total	39 (41 %)	8 (8.4 %)	10 (10.5 %)	2 (2.1 %)	2 (2.1 %)	0 (0 %)	8 (8.4 %)	5 (5.2 %)	21 (22.1 %)

- Risk: Knowledge related to risk addresses different threats that might threaten the assets of an organization, the vulnerabilities that might be explored, the attackers, and their attack methods.
- Risk treatment: knowledge related to mitigating the risk, such as security requirements, countermeasures, and security policies.

4.2.2 Form of representation

The "form of representation" facet indicates the different types of knowledge forms that were identified and how they are organized: patterns, taxonomies and ontologies, templates and profiles, catalogs and generic models, and mixed.

4.2.3 Technique

The technique facet defines whether the knowledge (re)use techniques can be automated (e.g., queries), semi-automated (e.g., process), or are totally manual (e.g., guidelines).

4.2.4 Automated support

The "automated support" facet checks the existence of an automated support for knowledge (re)use in SRE and its technology features.

The next section details the publications retrieved and replies to the research questions relying on the presented framework.

4.3 Details of the systematic mapping study

Table 2 presents an overview of the security knowledge reuse in SRE methods. Columns contain the main concepts characterizing the conceptual space of security. Lines cover the different reuse forms by SRE methods. Cells contain a colored area when there exist SRE publications proposing a reuse-based approach of a given reuse form, for a given security concept. The colors of the cells get darker according to the number of publications covering it. It is white when there is no publication describing such a link.

This presentation should help the reader to understand the security reusable knowledge in the body of literature. It also helps to retrieve for each concept of security (e.g., security requirement) how (through ontologies, templates) and how much it is reused. As an illustration, the security concept "threat" is covered by a lot of publications proposing to reuse it through ontologies or taxonomies.

The following paragraphs go into the details. They present a brief description of the SRE method, followed by

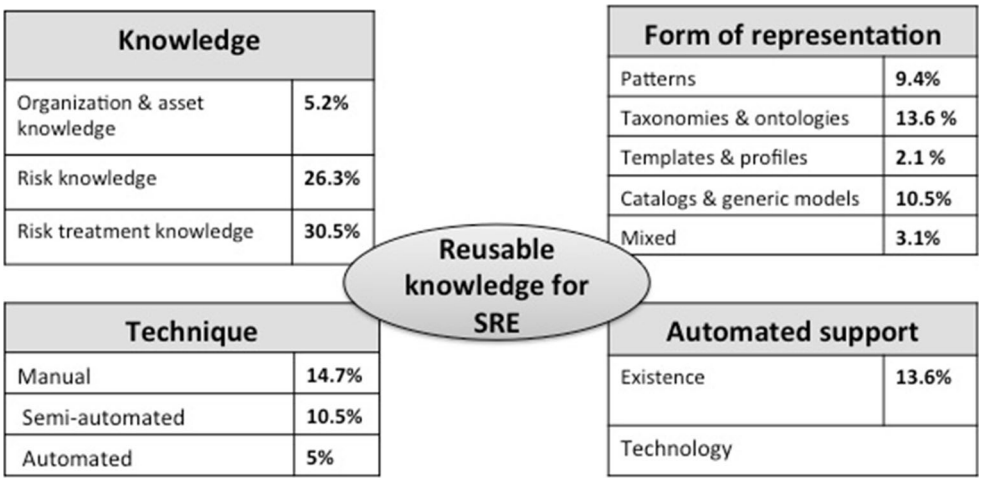

Fig. 4 Framework for knowledge reuse in SRE

Table 2 Security knowledge reuse in SRE

	Organization	Asset	Threat	Vulnerability	Security goal	Security requirement	Counter-measure
Ontologies/ taxonomies							
Catalog/ generic models							
Patterns							
Templates							
Mixed							

answers to the research questions (method, form of representation, technique, and automation). The paragraphs report the different types of reusable elements and how they are modeled and described, and how they are used.

In fact, there are different ways for presenting and classifying SRE methods, depending on the angle from which we study and analyze them. For instance, Fabian et al. [27] organize SRE methods into six main categories (multilateral, UML-based, goal-oriented approaches, problem frames-based, risk analysis-based, common criteria-based). Elahi [28] organizes SRE methods into two main categories depending on whether the method focuses on the threats and vulnerabilities (the dark side of security) or on security requirements and countermeasures (the white side of security).

As the main goal of this paper is to focus on knowledge (re)use in SRE, the different methods will be presented using the result of RQ2, i.e., according to the knowledge (re)use form used. Thus, we distinguish: methods that reuse patterns, taxonomies and ontologies, catalogs or generic models, and mixed forms of reuse. We also distinguish methods that do not reuse any kind of security knowledge.

4.3.1 Methods that (re)use security patterns

A security pattern describes a particular recurring security problem that arises in specific contexts and presents a well-proven generic scheme for its solution [29]. The SMAP found that some SRE methods (re)use patterns during the SRE process in the form of models or in other forms.

4.3.1.1 Patterns of models The identified SRE that (re)use patterns of models are presented below:

- *KAOS and Anti-Models:* Lamsweerde [5, 30] extended the KAOS method to support security issues at the

requirements level. KAOS is a requirements engineering method dealing with the elaboration of the objectives to be achieved by the system-to-be. **(Knowledge reliance)** Hermoye et al. [11, 31] enriched the KAOS framework with an attack pattern library and reusable countermeasures built after analyzing commonalities in goal-oriented specifications from some case studies. **(Reusable Knowledge)** In this approach, a reusable attack pattern captures common objectives of malicious agents for known attacks (e.g., replay, denial of service, and password attacks). Reusable countermeasures are reusable anti-goal resolutions. For example, countermeasures against replay attacks may include freshness mechanisms. **(Form of representation)** Hermoye et al. use a library of attack patterns; an attack pattern is a fragment of an anti-model defined on an abstract domain. Attack patterns are built with abstract *anti-goals* and abstract *domain properties.* Abstract *anti-goals*, *domain properties,* and predicates are reusable concepts defined on abstract domains that should be specialized in concrete domains at reuse time. **(Technique)** Hermoye et al. provide a formal technique to reuse this library for threat analysis. They propose three main functions: *Retrieve* to get initial anti-goals, and *Specialize* and *Adapt* to specialize each abstract variable (e.g., objects, agents, and relations) of the attack pattern. **(Automation)** The KAOS method is supported by the Objectiver[1] tool. Even though we do not have details about technical aspects, the tool offers some functionalities such as modeling requirements and related concepts (goals, obstacles, expectations, hypotheses, etc.), querying the model to retrieve some model elements, exporting in XML format, and data exchanges in XMI format. Note that the tool does not handle the reuse of knowledge for SRE.

- *Secure Tropos:* secure tropos method is derived from Tropos. The latter is a software development method based on the paradigm of agent-oriented software development [32, 33]. There are different extensions of Tropos in the literature. Mouratidis et al. [4, 34, 35] extend Tropos with new concepts to cover security modeling (security constraints, secure dependencies, and secure entities) and more. secure tropos distinguishes four main development phases: early requirements, late requirements, architectural design, detailed design and architectural design. Recently, secure tropos was extended to be used in the field of cloud computing [36–37]. **(Knowledge reliance)** In a previous work, Tropos method was extended with security patterns [38]. **(Form of representation)** Authors describe a pattern language, based on agent-oriented concepts. They used the Alexandrian format [39] for organizing each pattern. In this format, the sections of a pattern are *context*, *problem and forces*, *solution*, and *rationale*. **(Reusable knowledge)** Authors proposed four main patterns: "Agency Guard" concerned with ensuring that there is only a single point of access to the agency to protect it from malicious agents. "Agent Authenticator" related to authentication of agency's agents. "Sandbox" related to mechanisms for separating running activities. "Access Controller" suggests intercepting all requests for the agency's resources. **(Technique)** Mouratidis et al. provide some guidelines and show how these patterns can be integrated within the architectural design stage of the Tropos agent-oriented methodology. **(Automation)** ST-Tool is one of the main tools known for secure tropos. Formal analysis is based on logic programming. ST-Tool [40] provides a graphical user interface (GUI) that allows designers editing secure tropos models as graphs where nodes are actors and services, and arcs are relationships. To the best of our knowledge, ST-Tool does not handle the development using patterns for elicitation.

4.3.1.2 Patterns not models The identified SRE that (re)use patterns are presented below:

- Okubo et al. [41] propose a method for security impact and security requirements analyzes. There are two types of security impact described with more details in the paper: *horizontal impact* on artifacts in the same stage and *vertical impact* on artifacts in a later stage. **(Knowledge reliance)** The method proposed by Okubo et al. consists of two techniques: an analysis method of horizontal impacts using an extended misuse case; a combination of new security patterns and a traditional traceability technique to analyze security vertical impacts. The security patterns bridge the gap between security requirements and the design, so as to know impacts on code when security requirements change. **(Form of representation)** Okubo et al. constructed security requirements patterns (SRPs) and security design patterns (SDPs). A security requirement pattern is formed around a "context," a "problem," a "solution," and a "structure." In addition, a security design pattern has: "consequences," "implementation," and "sample code." **(Reusable knowledge)** In terms of knowledge, SRPs provide assets and threats. SDPs provide countermeasures. **(Technique)** The authors propose a process for security impact analysis that starts with selecting the SRP, identifying new assets, identifying new threats, identifying countermeasures, and finally, selecting the SDP, and ends with estimating the

[1] http://www.objectiver.com/.

impact for each countermeasure. **(Automation)** The method is not tool supported.

4.3.2 Methods that (re)use taxonomies or ontologies

An ontology is a formal representation of the entities and relationships which exist in some domain. A taxonomy is an ontology in the form of a hierarchy. Whereas ontologies can have any type of relationship between categories, in a taxonomy there can only be generalization hierarchies [42]. The SMAP revealed a variety of SRE methods that suggest the use of ontologies or taxonomies during a SRE process:

- *GBRAM*, the goal-based requirements analysis method [43, 44] is a straightforward methodical approach to identify system and enterprise goals and requirements. **(Knowledge reliance & Form of representation)** Anton et al. [45] propose a requirements taxonomy for reducing Web site privacy vulnerabilities. They evaluated 25 Internet privacy policies from eight non-regulated e-commerce industries. The evaluation permitted us to identify main goals and vulnerabilities. (**Reusable knowledge)** The security knowledge in the taxonomy was categorized into *Privacy Protection Goals* and *Privacy Vulnerabilities*.
  - *Privacy protection goals* express the desired protection of consumer privacy rights. They were categorized into five categories: *notice/awareness, choice/consent, access/participation, integrity/security,* and *enforcement/redress.* For instance, *notice/awareness* goals assert that consumers should be notified or made aware of an organization's information practices before any information is actually collected from them. More details about the other goals can be found in Aton et al.'s publication [45].
  - *Privacy vulnerabilities* reflect ways in which a Web site may violate consumer privacy. The seven main categories of privacy vulnerabilities are: *information monitoring, information aggregation, information storage, information transfer, information collection, information personalization,* and *contact.*

  (Technique) The authors mentioned that Web site designers can use this taxonomy to ensure that their stated and actual policies are consistent with each other and it can be used by customers to evaluate and understand policies and their limitations. However, there were no precise procedures or techniques for the use. **(Automation)** As far as we could determine, the GBRAM method is not tool supported.
- *Secure Tropos:* Another extension of the Tropos methodology was the one proposed by Massacci et al. [2, 40, 46–48]. The authors use the secure i* (Si*) language. In addition to the notions originally supported by the i* modeling framework, Si* introduces the notions of *delegation* and *trust*. *Delegation* is defined as a relation between two actors (the *delegator* and the *delegatee*—the one to whom something is delegated) and a goal, task, or resource (the *delegatum*). The notion of *trust* is used to separate delegation between trusted and untrusted actors. Similar to delegation, trust is defined as a relation between two actors (the *trustor* and the *trustee*) and a goal, task, or resource (the *trustum*). A third extension to Tropos was proposed by Asnar et al. [49, 50] for risk modeling, the tropos goal risk framework, to assess risk, based on trust relations among actors. **(Knowledge reliance)** Massacci et al. [51] propose a formal ontology for socio-technical systems. **(Form of representation)** Authors formalized the concepts of Si* into an ontology. **(Reusable knowledge)** The concepts are organized into *extensional* and *intentional* predicates. *Extensional predicates* correspond to the edges and circles drawn by the requirements engineer (e.g., service, goal, task, and resource) during the modeling phase. These predicates are used to formalize the intuitive description of the system. *Intentional predicates* are determined with the help of rules by the reasoning system; examples of these predicates are: `aim(Actor:x, Service:s)` and `has_perm(Actor:x, Service:s)`. **(Technique)** The authors provide some axioms that define the semantics underlying Si*. They are used to complete the extensional description of the system. All these primitives were used to deal with the security organizational requirements.
  The proposition (`Pro`) in the example below verifies whether an actor (X) who delegates the permission (`perm`) to another actor (`Y`) to deliver a service (`S`) is entitled to do it. With other predicates, one can verify the authorization security requirement.
  For example, `Authorization: Pro` ← `delegate(perm, X, Y, S)` ∧ `not has_perm(X, S)`.
  (Automation) As far as we know, there was no automation support for this secure tropos extension.
- *RITA:* Elena Ivankina et al. [52, 53] **(Knowledge reliance)** present a requirements elicitation method called requirements identification threat analysis (RITA) that makes use of a threat ontology. **(Form of representation)** Security requirements in RITA are expressed in forms of treatment that prevent threats. Treatments are formalized as goals. A goal is defined as "something that some stakeholder hopes to achieve in the future" [54]. A goal is expressed as a clause with a main verb and several parameters, where each

parameter plays a different role with respect to the verb. Example of a security requirement in RITA (treatment): "provide (connection help) object (to users) destination (when the connection fails) time." In this example, the parameters of the verb "provide" are: the object (connection help), the destination (to users), and the time (when the connection fails).
(Reusable knowledge) The threats ontology in RITA organizes types of threats into classes and subclasses at several levels. Five classes are defined on the highest level: "user," "design," "environment," "hardware," and "engineering." Classes and subclasses are characterized by distinctive variables that help identify threats in the ontology and define each class distinctively from the others. RITA also uses a second ontology that proposes a series of generic treatments for the generic threats identified in the threats ontology. **(Technique)** RITA further relies on some additional guidelines to use the ontologies: It uses a matrix in which each threat in the threat ontology corresponds to the appropriate treatment from the treatment ontology. **(Automation)** RITA was implemented with a prototype whose main function was to demonstrate the implementability of the method.

- Daramola et al. [55, 56] present an approach that leverages ontologies and requirements boilerplates in order to alleviate the effects of the lack of inexperienced personnel for security requirements specification (SRS). **(Knowledge reliance)** Daramola et al.'s approach makes use of ontologies and requirements boilerplates. **(Form of representation)** A requirement boilerplate [57] is a pre-defined structural template for writing requirement statements. The fixed parts of requirement boilerplate are reused when writing requirements, while the requirement engineer can manually fill in the parameter parts with information from its application.
  An example of a boilerplate:

  "BP2: The <system> shall be able to <action> <entity>"

The ontologies provides the necessary knowledge that is required to identify security threats and recommend appropriate countermeasures, while the requirements boilerplates provide a reusable template for writing security requirements in a consistent way in order to eliminate ambiguity. **(Reusable knowledge)** The basic threat ontology (BTO) used in the approach contains a mapping of some kinds of security threats to specific defense actions based on information that was gathered from the literature and existing security ontologies. **(Technique)** The knowledge contained in the BTO is used for automatic recommendation of appropriate defense actions. This is made through ontology reasoning and other semantic capabilities. **(Automation)** The proposed approach is tool supported by the prototype ReqSec tool. ReqSec is an eclipse-based tool that provides automated support for ontology-based security requirements specification by enabling the specification of security requirements from textual misuse case descriptions.

- Dritsas et al. [58] introduce **(Knowledge reliance)** a knowledge-based approach for the security analysis and design of e-health applications. **(Form of representation)** The authors describe different threats to security within the e-health applications domain. They then draw a table of different security requirements (authentication, authorization…) with description of each in the e-health context. Finally, they specify these requirements embedded in a set of patterns. Each pattern contains a name, overview, problem, solution, requirements, asset, threats, vulnerabilities, and related patterns. Note that the authors did not present any evaluation of their proposed approach. **(Reusable knowledge)** Dritsas et al. made use of a security ontology for e-health applications. In this ontology, the concept of a security pattern is a representation of the security patterns and is connected with the concept of countermeasure through a *provide* relationship: Each security pattern provides a specific set of countermeasures. In practice, each security pattern is matched with a set of countermeasures during the ontology instantiation. A security pattern context is defined as a set of asset, vulnerability, and deliberate attack triplets. In this way, one can start from the generic security objectives, find the Security Pattern Contexts that match them and, thus, choose specific Security Patterns. Therefore, the high level security requirements and objectives can be fulfilled by implementing the respective countermeasures. **(Automation)** The approach is not tool supported. **(Technique)** The following query (Q) illustrates how the ontology is used by a developer involved in an e-health development project. The query (Q) asks what are the countermeasures to consider in order to protect the medical data of a patient. The result returned by the query (from the ontology) suggests considering the countermeasures: encryption, access control, certificates, intrusion detection, and malicious software detection.
  Q. What countermeasures protect the medical data of a patient?

```
nRQL Query: (retrieve (?cm) (|Medical_Data| ?cm |protected_by|))
nRQL Result:
 (((?CM |Encryption|))
 ((?CM |Access_Control|))
 ((?CM |Certificates|))
 ((?CM |Intrusion_Detection|))
 ((?CM |Malicious_SW_Detection|)))
```

- Lasheras et al. [59] propose an ontological representation for reusable requirements, which allows incompleteness and inconsistency in requirements to be detected and semantic processing in requirements analysis to be achieved. Note that the framework seems to be at an early stage, in the sense that it does not permit security requirements elicitation and analysis. To date, its contribution is limited to the proposed ontologies. **(Automation)** The framework is not supported with any tool **(Knowledge reliance & form of representation)** Lasheras et al. defined some reusable knowledge encapsulated in ontologies. **(Reusable knowledge)** Authors defined two kinds of ontologies: a risk analysis ontology and a requirements ontology.
  - The risk analysis ontology is based on MAGERIT [60], the information systems risk analysis and management method of the Spanish public administration. The ontology identifies five types of risk elements: *Asset*, *Threat*, *Safeguard*, *Valuation dimension, and Valuation criteria*.
  - The concepts, meta-information, and relationships included in the requirements ontology have been mostly taken from the authors' experience of requirements reuse-based method SIREN [61]. The ontology organizes requirements into *software requirements* and *system requirements*.

(Technique) Lasheras et al. [59] rely on semantic queries on the ontology to verify the correctness of the requirement.

- *Salini* et al. [62] introduce **(Knowledge reliance)** a knowledge-oriented approach addressing the security requirements engineering phase for developing an e-voting system. **(Form of representation)** For the knowledge part, the authors provided a security requirements ontology for e-voting systems. **(Reusable knowledge)** The terms used as ontology classes are the following: stakeholder, security objective, threat, security requirements, assets, vulnerabilities, security requirements pattern, impact, severity, and web application. The relations among the ontology classes and the properties used to represent the relations are use, have, requires, is vulnerable to, implemented in, protects, mitigated by, provide, damage, affects, exploited by, addresses, assessed and part-of. Salini et al. explained that in practice, each security requirements pattern is matched with a set of security requirements during the ontology instantiation. A security requirements pattern is defined as a set of asset, vulnerability, threats, and impacts. In this way, one can start from the security objectives, find the security requirements pattern that matches them and, thus, choose specific security requirements. Although the approach seems to be interesting and useful for defining security requirements, there was no validation reported for it, nor for the proposed security ontology. The ontology is still under development (not all identified security requirements have been mapped to the security objectives). **(Automation)** The approach is not supported with any tool.
- Chikh et al. [63] **(Knowledge reliance)** present a framework for building security requirement specifications related to information security requirements (ISRs) using ontologies. **(Form of representation)** The framework uses three kinds of generic ontologies as a solution to this problem—software requirement ontology, application domain ontology, information security ontology. However, despite the fact that the framework looks promising, it is difficult to know its usefulness, since no validation was presented. **(Reusable knowledge)** Chikh et al.'s framework uses the security ontology proposed by Fenz et al. [64]. This ontology encompasses security knowledge related to threats, vulnerabilities, controls, organization, and its assets. **(Automation)** The authors mentioned ongoing development of a prototype to evaluate their proposition.

4.3.3 Methods that (re)use templates or profiles

Some SRE methods rely on templates and profiles as another kind of reusable knowledge for SRE. The identified methods that (re)use this forms are:

- Zuccato et al. [65] present an approach that organizes security requirements engineering around five activities. The first activity starts with a simplified risk analysis approach by means of questionnaires to identify areas in the business which can have security problems. Subsequently, the security requirements for

the development project are selected (requirement profiles). These requirements are then forwarded to the suppliers. **(Knowledge reliance & form of representation)** The method proposed by Zuccato et al. is based essentially on the use of security requirements profiles that address a business domain, in a commercial organization, where activities have to serve a business purpose (not to be confused with security patterns which describe a security domain solution according to authors). **(Reusable knowledge)** Typical examples for this business orientation would be IPTV Services (e.g., renting a movie, recording some programs, delayed viewing,…), VoIP (e.g., multiple numbers, location locking, answering machine,…), or customer self-administration (e.g., mypages, myworkingpages, MyFamily, mobile device management (MDM)…) where a profile is created for the service category and then reused, with some adaptation, for the specific service. **(Automation)** The approach is not tool supported.

- *Firesmith* [10] **(Knowledge reliance & Form of representation)** suggests using textual security requirements templates (not to be confused with security use cases templates). An example of a reusable parameterized template for specifying an integrity security requirement:

  > "The [application center/business unit] shall protect the data it transmits from corruption (e.g., unauthorized addition, modification, deletion, or reply) due to [unsophisticated/somewhat sophisticated/sophisticated] attack during execution of [a set of interactions/use cases] as indicated in [specified table]."

Users of these templates can manually replace the brackets according to their different applications. **(Reusable knowledge)** More detailed templates in Firesmith's research could not be found. The proposition of the author is limited to the importance of specifying the knowledge into this kind of templates. **(Technique)** The author provides an asset-centered and reuse-based procedure for requirements and security teams to analyze security requirements containing 13 steps, starting by identifying the valuable assets, identifying threats, and estimating vulnerabilities. The steps end by specifying requirement through the instantiation of templates based on the parameters from the previous steps. **(Automation)** This proposition was not tool supported.

4.3.4 Methods that (re)use catalogs or generic models

Some SRE methods define generic models of common security problems and their solutions, in order to (re)use them. Some others rely on catalogs to encapsulate the reusable knowledge as presented below:

- *Misuse Cases:* Sindre and Opdahl [6, 66–68] extend the traditional use case approach to also consider misuse cases, which represent behavior not wanted in the system to be developed. Misuse cases are initiated by misusers. They have two representations: a graphical diagram and a textual specification. **(Knowledge reliance)** Misuse cases were initially developed without relying on any kind of knowledge repositories. However, Sindre et al. [69] then defined an approach based on a repository of generic misuse cases (generic threats and generic security requirements). **(Form of representation)** Sindre et al. represent the reused knowledge using generic misuse cases. Each misuse case has a name, summary, preconditions, misuser interactions, systems interactions, and postconditions. **(Reusable knowledge)** Authors suggest two main reusable artifacts: Generic threats (e.g., spoofing, i.e., a misuser gaining access to the system by pretending to be a regular user) and generic security requirements (e.g., access control) described independently of particular application domains. **(Technique)** Authors provide a way to use/reuse this repository through some guidelines. **(Automation)** As far as we know, misuse cases are still not tool supported.
- *Abuse frames.* Lin et al. [70–72] define so-called anti-requirements and the corresponding abuse frames. Their proposition is comparable to problem frames introduced by Jackson [73]. An anti-requirement specifies the undesirable phenomena in the system that must be prevented from happening; it expresses the intentions of malicious users. An abuse frame represents a security threat. Authors incorporate anti-requirements into abuse frames to represent a security threat. The authors state that the purpose of anti-requirements and abuse frames is to analyze security threats and derive security requirements.

(Reusable knowledge) In problem frames, each frame describes a particular problem class (e.g., information display, workpiece, and required behavior frames). Similarly, Lin et al. propose abuse frames that describe classes of security violation (interception, modification, and denial of access). Each one represents a threat that can violate a particular security goal. **(Knowledge reliance & Form of representation)** These security violations, represented through abuse frames diagrams, are meant to be reusable. As an example, Fig. 5 shows a standard modification frame. Modification arises whenever an attacker wishes to change an information asset in the physical world. The problem is to find a modification machine that allows an attacker to achieve it. Modification violates

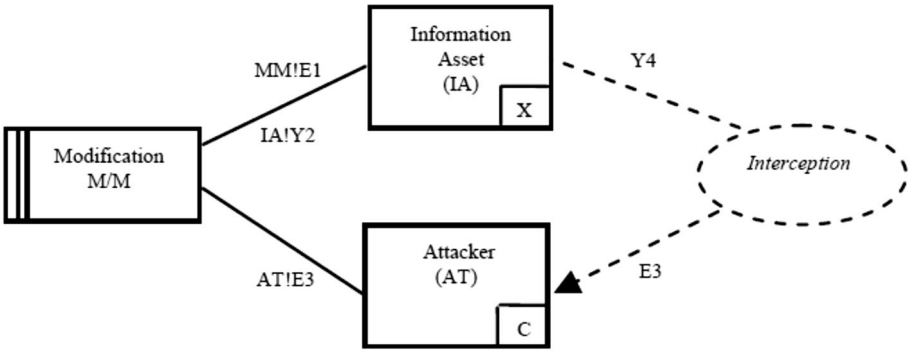

Fig. 5 A standard modification abuse frame taken from [70]

integrity. **(Technique)** The authors propose an iterative threat analysis method that essentially comprises four steps (scoping the problem and identify the sub-problems, identifying the threats and constructing abuse frames, identifying security vulnerabilities, and addressing security vulnerabilities). **(Automation)** As far as we know, abuse frames are not supported with a tool.

- *Security Use Cases:* Donald Firesmith presented security use cases [7]. **(Knowledge reliance)** Firesmith tried to keep the security use cases templates [7] at a reasonably high level of abstraction for reusability purposes. **(Form of representation)** Security use cases have a name, a path, a security threat, preconditions, misuser interactions, system requirements, and post-conditions. **(Reusable knowledge)** The author presented the reusable use cases: access control, integrity, and privacy **(Technique)** The author emphasizes that, when reused on real projects, each path template can be made more specific to the application by replacing the general words "system" and "user" with the specific application name and the specific type of user. **(Automation)** To our knowledge, security use cases are not tool supported.
- Saeki and Kaiya [74] propose a weaved security requirements elicitation method that uses **(Knowledge reliance)** common criteria (CC) [75] and related knowledge sources to identify security requirements from functional requirements through eliciting threats and security objectives. **(Reusable knowledge)** The authors think that CC can be considered as a kind of catalog to provide knowledge on threats, security objectives, and security functions that have generally appeared. For example, by using common criteria, one can select the objective "data encryption" from the catalog, to mitigate the threat "disclosure of password data." **(Technique)** The proposed technique is to weave through CC two types of requirements elicitation: One is any existing functional requirements elicitation and the other is a typical method for eliciting security functional requirements. **(Form of representation)** The method relies on CC as a source of knowledge to support activities of security requirements elicitation. The authors used CC Part 2 [76], which has about 120 Security Functional (SF) components, as a catalog. In addition, they used ECMA-271 E-COFC [77] (which can be considered as a profile of CC in a certain problem domain), as catalogs of threats and security objectives. As shown in the right-hand side of Fig. 6, the method accumulates a threat catalog, a security objective catalog, and a SF component catalog and holds relationships between their catalog entries (i.e., security objective mitigates threat and SF component represents security objective). **(Automation)** As far as we know, the proposition of Saeki and Kaiya is not tool supported.
- *SIREN for security requirements*. Toval et al. [61] propose an approach for security requirements elicitation. **(Knowledge reliance)** The approach is a particularization of SIREN (SImple REuse of software requiremeNts), a general-purpose RE method based on requirements reuse. The particularization of SIREN to the security profile has been based on the risk analysis and management method MAGERIT [60]. Security requirements specify the countermeasures prescribed by MAGERIT after the risk analysis. Therefore, it is the MAGERIT risk analysis and management that determines the security mechanism to be used in each circumstance. SIREN encompasses a process model and some guidelines. **(Form of representation)** The guidelines that SIREN provides consist of a *hierarchy of requirements specification documents* together with the templates for each document. These serve to structure a *reusable requirements repository*. **(Reusable knowledge)** The repository defined in SIREN contains functional and non-functional requirements from specific domains and profiles. A SIREN profile consists of a homogeneous set of requirements that can be applied to a variety of domains, such as information systems security and the personal data privacy law. There are two main types of requirements in the repository:

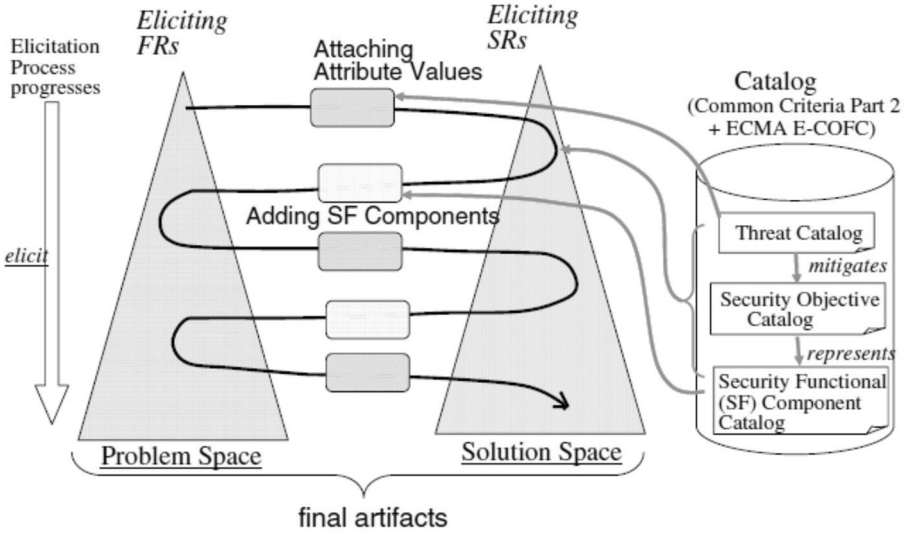

Fig. 6 Using knowledge included in common criteria taken from [74]

- Parameterized: This kind of requirements contains some parts that have to be adapted to the application being developed at the time. If this requirement is chosen, the parameterized part will be instantiated, that is, the information in parentheses will be replaced by a specific value according to the current project. For example: "SRS.3.5.3.1.S.301 The security manager shall check the user's identifiers every [time in months] to detect which ones have not been used in the last [time in months]."
- Non-parameterized: requirements that could be applied directly to any project concerning the profiles and/or domains in the repository. For example: "SRS.3.4.3.S.5. The firewall configuration will be screened host."

(Technique) Toval et al. adapted a spiral life cycle in SIREN to take requirements reuse explicitly into account in the RE process. Details about the process can be found in [61]. **(Automation)** To our knowledge, SIREN is not tool supported.

- *Secure Tropos:* **(Knowledge reliance)** In their recent work [35, 36, 78], Mouratidis et al. suggest considering the activity of cataloging during the elicitation and analysis process. The main aim of this activity is to develop a reference catalog model that can be employed not just for the project for which it was initially developed but can work as a reference model for any projects that demonstrate similar characteristics. **(Form of representation)** The reference catalog diagram takes the form of a reference model that contains graphical representation of different concepts needed for elicitation process. **(Reusable knowledge)** The reference catalog provides relationships between the concepts security and privacy goals, threats, security and privacy measures, and security and privacy mechanisms. For example, the security goal "availability" can be threatened by the threats "Data Location" and "Insecure Storage." The measure to mitigate these threats can be "API Interoperability" which uses the mechanisms "middleware, support multiple providers." For their case study, authors used existing information in the security document of the company to construct a cataloging diagram. **(Automation)** The framework is supported by a tool, which has been developed based on the Open Models Initiative ADOxx Platform.[2] The tool provides an environment for developers to create a number of diagrams that support the process of the method. In particular, the tool permits development of the Security and Privacy Reference Catalog Diagram discussed before.

4.3.5 *Methods that (re)use mixed forms of security knowledge*

There are a variety of SRE methods that (re)use different (mixed) forms of knowledge; our SMA identified the following ones:

- *SQUARE* [79–81] is a multilateral approach. Multilateral security [82] aims at a balance between the competing security requirements of different parties. SQUARE aims to integrate security requirements engineering into software development processes [80]. **(Knowledge reliance)** Travis et al. introduced a new variant of SQUARE, R-SQUARE [83], which is defined using SQUARE Lite as a base model and

[2] www.openmodels.at.

incorporating reuse in some places of the process. **(Reusable knowledge)** However, the introduced layer of reusable knowledge gives only some indications and no more. Throughout the selected publications, it was not possible to access to this reusable knowledge. For example, during the "agree on definitions" step, the authors suggest creating and maintaining a glossary of relevant terms and definitions so that the meanings of requirements do not become ambiguous over time as they are reused. During the identification of assets and goals step, the authors recall that organizations that develop product lines of secure software [84] will likely have overarching business and security-related goals that are intended to apply to all affected projects. During the risk assessment phase, R-SQUARE method suggests to use threat models, which are known to be abstract and highly reusable. **(Form of representation)** As understood from papers related to R-SQUARE, the method uses mixed forms of representation of the reusable knowledge such as definitions, models, and product lines. However, the authors did not provide any example to better understand what these forms look like. **(Technique)** It is not clear what the techniques used to access the reusable knowledge are. **(Automation)** SQUARE has been automated by means of the P-SQUARE tool; this tool is designed for use by stakeholders, requirements engineers, and administrators. It supports both the security and privacy aspects of SQUARE by recording definitions and searching and adding new terms, identifying the project business goals, assets, and security or privacy goals, adding or editing links to project artifacts performing risk assessment and identify threats. No technical details were provided concerning the tool. Moreover, the tool P-SQUARE does not support R-SQUARE (Reusable SQUARE).

- *SREP.* Mellado et al. [85, 86] present the security requirements engineering Process (SREP). SREP is an iterative and incremental security requirements engineering process, which is based on the unified process [87] software life-cycle model with multiple phases. **(Knowledge reliance)** SREP is asset based, risk driven, and following the common criteria (CC) [86], it supports the reuse of security requirements, as well as the reuse of knowledge on assets, threats, and countermeasures. **(Form of representation)** It relies on a security resources repository (SRR), which stores some reusable security elements that are of different forms: plain text, security use cases, attack trees, and misuse cases. **(Reusable knowledge)** The meta-model showing the organization of the SRR is exposed in Fig. 7. The most important aspects of it are:
  - Generic Threat and Generic Security Requirements are described independently of particular domains.
  - Security Requirement Cluster is a set of requirements that work together in satisfying the same security objective and mitigating the same threat.
  - The Req–Req relationship allows an inclusive or exclusive trace between requirements. An exclusive trace between requirements means that they are mutually alternative, as for example that they are in conflict or overlapping. Whereas an inclusive trace between requirements means that to satisfy one, another (others) is (are) needed to be satisfied.

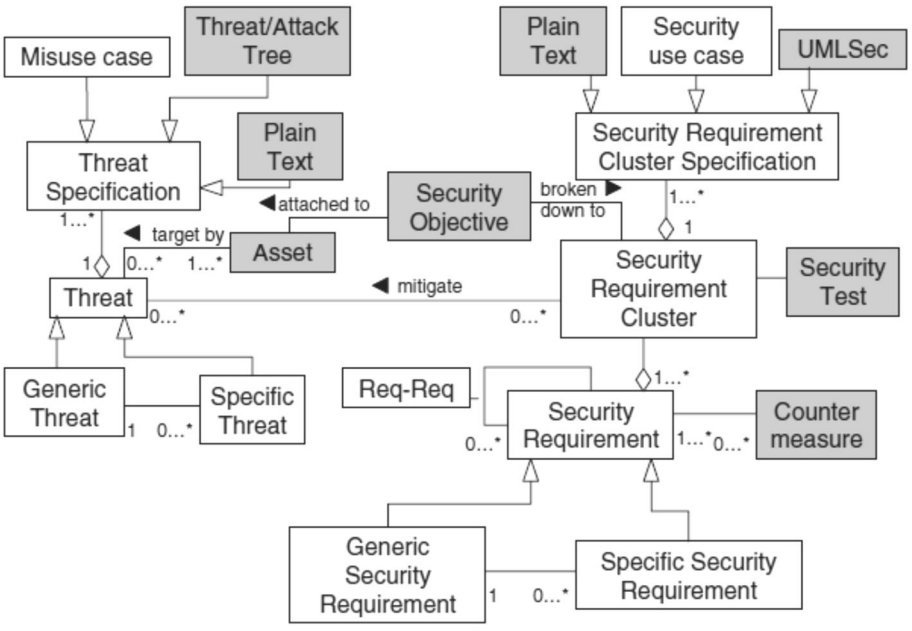

Fig. 7 Meta-model for security resources repository taken from [86]

(Automation) Tool support is critical for the practical application of the SREP in large-scale software systems due to the number of handled artifacts and the several iterations that have to be carried out. However, the authors have not developed it so far.

4.3.6 Methods that do not (re)use security knowledge

The SMAP found that there are a wide variety of SRE methods that do not consider knowledge reuse during the SRE process; we summarize them below. Note that for this category, we obviously skip answering the questions related to reusable knowledge, form of representation, and technique. Since there is no knowledge reliance, we cannot talk about the points related to the knowledge reused, its form of presentation, or the technique for reusing it.

- *Secure I*:* Liu et al. [3, 88, 89] propose employing explicit modeling of relationships among strategic actors in order to elicit and analyze security requirements. Authors analyze attackers, vulnerabilities in an actor's dependency network, countermeasures, and access controls. In Liu et al.'s contribution, actors are assumed to be potential attackers that inherit capabilities, intentions, and social relationships of the corresponding legitimate actor. **(Knowledge reliance)** Authors mentioned that it would be useful to retrieve attacks and prototypical solutions from pre-defined taxonomies or knowledge repositories [3], but the method, as it is, does not handle the use of this kind of knowledge so far. **(Automation)** Secure I* was not initially tool supported [3]. Later, Giorgini et al. adapted secure I* concepts within secure tropos and proposed ST-tool [40] which is used today for secure I* diagrams. As far as we know, the tool does not support reuse of knowledge for SRE.
- *UMLSec* and *SecureUML* are two main UML-based extensions for modeling security. SecureUML [8, 90] is a UML-based modeling language for the model-driven development of secure systems [8]. SecureUML takes advantage of role-based accesscControl (RBAC) for specifying authorization constraints by defining a vocabulary for annotating UML-based models with information relevant to access control. Using UMLsec [9, 91–93], security requirements are defined by assigning security stereotypes, constraints, and tagged values, which are defined in a UML profile for the elements of the design models. **(Knowledge reliance)** Neither UMLSec nor secureUML considers the (re)use of security requirements knowledge.
- **(Automation—SecureUML)** Araujo et al. [90] present a secureUML template—a Microsoft Visio template built to model authorization systems. The tool allows architects to model their role-based access control systems. We could not find technical information about the secureUML template. According to Araujo et al., the proposed template helps developers by identifying poor authorization design and implementations, helping to find contradictions/holes such as backdoors, or identifying authorization bypass opportunities.
  (Automation—UMLSec) Jürjens et al. [92] present a framework for verification of UMLsec models for security requirements. The framework provides three input and output interfaces for the analysis plug-ins: a textual command-line interface, a graphical user interface, and a web interface. Inputs can be UML diagrams in the form of XMI files, as well as textual parameters. As output can be UML diagrams such as XMI (or.zuml) files and text messages, advanced users of the UMLsec approach can use the tool to implement verification routines for the constraints associated to self-defined stereotypes. A new UMLSec implementation variant called CARiSMA [94] has existed since 2012, and this time is based on the Eclipse Modeling Framework. CARiSMA enables users to perform compliance analyses, risk analyses, and security analyses.
  (Reusable knowledge) Neither the automation for UMLSec or for secureUML supports the reuse of knowledge.
- *CORAS*: Dahl et al. [95–98] present an organizational model-based method that covers threat, vulnerability, and security risk analysis. It also covers the elicitation of security goals. The language consists of five different kinds of diagrams: *asset diagrams*, *threat diagrams*, *risk diagrams*, *treatment diagrams,* and *treatment overview* diagrams. Their basic building blocks are presented in Fig. 8.
  (Knowledge reliance) We could not find any papers that present the CORAS method (re)using security knowledge. **(Automation)** The CORAS Tool [97] follows a client–server model and is developed entirely in Java. The CORAS client application permits the analyst to create new analysis projects and documents, to edit security analysis results, and to generate analysis reports. The latest version [96] has in addition a user interface containing a pull-down menu, a tool bar, and a palette that contains all model elements. The CORAS tool does not support the reuse of knowledge.
- *ISSRM*: Mayer et al. [1] propose a risk-based security requirements engineering framework that focuses on integrating risk analysis with requirements engineering activities. The main idea is to align Information Technology (IT) security with business goals. For this aim, the impacts of risks on business assets are analyzed; threats and vulnerabilities in the architecture are identified; and security requirements are defined in

Fig. 8 Basic building blocks of the CORAS diagrams taken from [98]

order to mitigate the risks. **(Knowledge reliance)** ISSRM approach does not rely on any kind of knowledge repositories. **(Automation)** ISSRM approach is not tool supported.

- *MORDA:* Evans et al. [99, 100] propose Mission-Oriented Risk and Design Analysis (MORDA) as a methodology for analyzing security risks. Morda combines threats, attacks, and mission impact concepts for deriving an unbiased risk metric. **(Knowledge reliance)** Through this literature review, no publication addressing security knowledge (re)use by MORDA was found. **(Automation)** MORDA is not supported by any tool.
- *CRAC++*: Morali and Wieringa present a method named CRAC++ [101], which is an extension of the older method CRAC [102]. The Confidentiality Risk Assessment and Comparison (CRAC) is an architecture-based method for confidentiality risk assessment in IT outsourcing. In CRAC++, the method is extended with a step to identify confidentiality requirements in outsourcing. In other words, the method specifies and identifies confidentiality requirements of the client that are not implied by the known confidentiality requirements of the provider, which therefore are candidates for inclusion in a Service Level Agreement (SLA) with that provider. Authors present a case study to evaluate the method. **(Knowledge reliance)** To the best of our knowledge, CRAC++ neither relies on nor uses predefined reusable structured knowledge. **(Automation)** CRAC++ is not equipped with a tool.
- *SREF:* Haley et al. [103, 104, 105] present a framework for security requirements elicitation and analysis called security requirements engineering framework(SREF). **(Knowledge reliance)** To the best of our knowledge, Haley et al. do not rely on knowledge reuse for their proposed SREF. **(Automation)** No tool is presented with this framework.
- *MSRA,* for Multilateral Security Requirements Analysis [106, 107], aims to apply the principles of multilateral security [82] during the requirements engineering phase of systems development. This is done by analyzing security and privacy goals of all the stakeholders regarding the system-to-be, identifying conflicts, and consolidating the different stakeholders' views. **(Knowledge reliance)** The method does not rely on reusable knowledge for security requirements elicitation. **(Automation)** No tool is presented with the method, to the best of our knowledge.

5 Summary

The systematic mapping study recalls a great interest in security requirements engineering with a considerable attention to the (re)use of knowledge for defining security requirements. This section returns to the main research questions of this paper and replies to them according to all the publications retrieved.

The following summarizes the answers to the research questions.

RQ1. Does the security requirements method rely on reusable knowledge?

Our results indicate that reuse knowledge is addressed in 29 (31 %) out of 95 papers. This allows us to conclude that overall, the deployment of reusable knowledge in security methods is relatively unexploited and possibly immature. The rate of evaluation papers found (only 10.5 %) indicates that most of the propositions are not evaluated regarding their applicability and usability in large-scale case studies. Moreover, most experiments do not involve end users from practice. One might say that this is due to the fact that most of these methods were proposed in an academic context, mostly through Ph.D. dissertations focusing on validating the proposition in a small-scale laboratory experiment rather than in large-scale case studies. Nevertheless, this indicates that more attention should be given to the applicability and usability of the deployment of knowledge (re)use in SRE.

RQ2. How is the reused knowledge represented?

Surveying the different proposals allowed us to identify different forms of knowledge representation: Patterns

constituted (9.4 %) of them, taxonomies and ontologies (13.6 %), templates and profiles (2.1 %), and catalogs and generic models (10.5 %). Few propositions (3.1 %) used a mix of these different forms. The rest of the proportion concerns proposals that do not reuse security knowledge. This gives us a picture about the different forms to represent and access knowledge proposed in the literature. The question remains of why some representations are more "popular" than others, and it would be interesting to find out more directly from academics and practitioners (through a survey) about why they may prefer some forms to others. For instance, one might suggest the hypothesis that ontologies are known to feature reasoning mechanisms, catalogs might be easy to access, and generic models might be easy to visualize for reuse.

In any case, the following summarizes the different forms of knowledge representation identified:

- **Security patterns:**
  - **Models:** Notably, the work of Hermoy et al. [11] who propose an attack pattern library containing attack trees using the KAOS framework, and the proposition of Mouratidis et al. [38] who enforce the secure tropos method with security patterns models.
  - **Not models:** The method proposed by Okubo et al. [41] makes use of security requirements patterns and security design patterns.
- **Generic models:** Some researches propose repositories of generic models for the purpose of reuse, such as generic misuse cases [69], security use cases [7], and abuse frames diagrams [72].
- **Security requirements templates (plain text):** Firesmith suggests reusable security requirements templates [10]. SIREN relies on a repository of parameterized and non-parameterized security requirements [61].
- **Ontologies:** Some approaches propose the use of ontologies for SRE [53, 55, 58–63]; most of them are in their early stages and not yet validated. In fact, existing categories of security requirements do not use these ontologies.
- **Taxonomies:** As a continuity of the proposed method GBRAM, Anton and Erap [45] propose a taxonomy for reducing Web sites privacy vulnerabilities.
- **Catalogs:** The recent work of Mouratidis et al. [78] suggested relying on a catalog of reusable models, but did not mention what these models contain exactly and how to use them. Saeki and Kaiya's [74] method makes use of common criteria catalogs that contain threats, security objectives, and SF components.
- **Profiles:** Zuccato et al.'s method uses what the authors call security requirements profiles [65].
- **Mixed:** The method SREP [85] relies on a security resource repository (SRR) which stores reusable security elements that can be represented in different forms (misuse cases, attack trees, security use cases, UMLSec, and plain text). The method R-Square [83] also uses different kinds of reuse structures such as definitions, glossaries, and threat models.

RQ3. What are the techniques for (re)using the knowledge?

Most of the approaches (14.7 %) provide manual guidelines for reuse; some of them add a process to follow. Few rely on semi-automated techniques (10.5 %) such as formal rules. The ontology-based approaches take advantages of reasoning features of ontologies. These results indicate a high tendency to reuse through manual guidelines and a low trend to automatic techniques (only 5 %), which can be seen as a weakness. By that, we mean that starting with a well-formalized knowledge source then reusing it through a human activity following some guidelines may lead to negative results if the process is not applied well.

RQ4. What are the main reused elements?

The main reused elements are often threats (26.3 %) and security requirements (30.5 %) (cf. Table 2). The reused knowledge might differ slightly from one approach to another, but there is always knowledge related to the dark side of security (threats) and the treatment side (security requirements). Reusing threats and security requirements (two important parts of a SRE process) is important and most proposed methods seem to be attentive to that as the results indicate. In fact, most methods propose the threats and the different security requirements that correspond to them (or mitigate them). However, let us recall that the scope of "security" is much larger than that. For instance, very few approaches reused knowledge related to the organization and its assets (5.2 %). So, what about the organizational side where threats appear and arise? (The assets to protect and their locations, the different persons involved in an organization, the organizational activities…). This knowledge can be reused too through different projects. In addition to threats, there are the attackers, or categories of attackers, their attack methods and their attack tools, classes of vulnerabilities and common impacts of threats. Research on reuse of knowledge in SRE should consider more elements of security and not just requirements and threats.

Figure 9 presents a conceptual graph containing two main levels. The first one (l1) represents the conceptual space of security. It contains the main concepts used in security and the relations between them; an organization has assets that

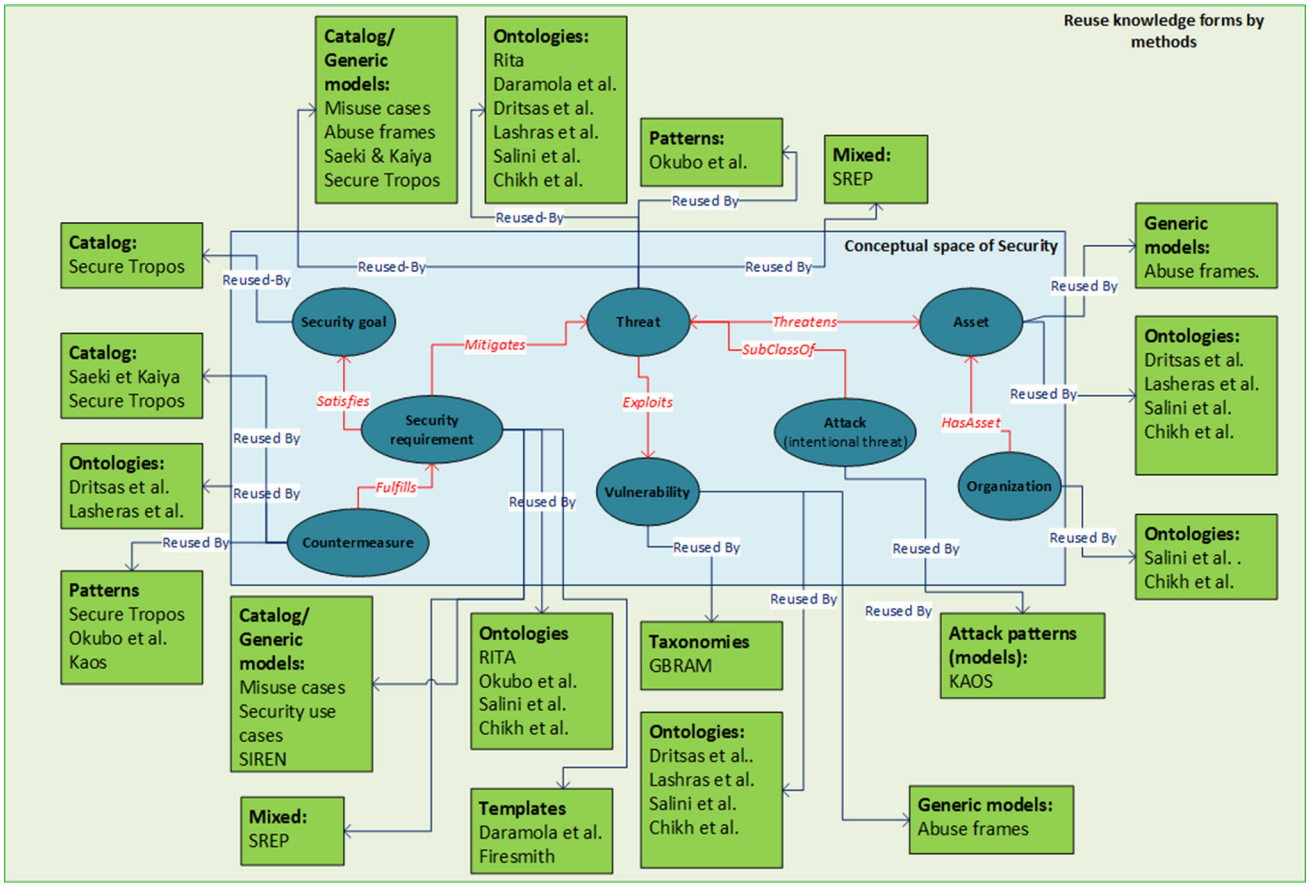

Fig. 9 Conceptual graph summarizing the knowledge reuse by SRE methods

threatened by threats. The latter exploit vulnerabilities and are mitigated by security requirements fulfilled by countermeasures. Security requirements satisfy security goals. This conceptual representation was adapted from the core security ontology presented in [108]. The second level (l2) represents the different knowledge forms and methods. The concepts of the two levels (l1 and l2) are related by the relation Reused-By. This presentation should help the reader to retrieve for each concept of security (e.g., security requirement) how it is reused (though ontologies and templates) and by which methods (Okubo et al., Firesmith).

RQ5. Are they tool supported?

Among the 95 selected papers, only 13.6 % propose tool support. However, most approaches do not provide tools that handle the *reuse of knowledge*, except one approach [56], where the authors present only a prototype. The tool mentioned by Mouratidis et al. [78] provides a way to create the reference catalog diagram and reuse it as discussed before.

This indicates that most propositions are unfortunately not tool supported. A possible explanation can be, as stated for RQ1, namely that in the academic environment where these methods were proposed, tool implementation is not the main focus.

RQ6. What are the new challenges regarding security knowledge (re)use in SRE?

Based on the SMAP presented in this paper, the challenges in the following are part of the authors' own view of open questions:

(Challenge 1) It is interesting to note that the risk-based approaches found do not handle reuse of security knowledge. The challenge will be to reconsider knowledge reuse in these methods. (Challenge 2) Many approaches for SRE relying on ontologies are emerging. They seem to be at their early stages and have not been validated yet. The challenge is to strengthen them and to apply them to large-scale case studies. (Challenge 3) Ontology-based approaches are not handled by the existing security requirements engineering categories (model based), and it would be interesting to see how to merge these two directions. (Challenge 4) There is a lack in automated support that handles knowledge reuse for the different SRE methods. More tools to support that would be appreciated. (Challenge 5) It would be interesting to generalize and

unify all these efforts (like in UML), so that they can more easily be exploited by companies.

Table 3 (in page 23) summarizes the results of the systematic literature review. The columns contain the different SRE methods grouped into categories. The lines contain the main aspects of knowledge reuse (form of representation, reusable knowledge elements, reuse technique, and automation). The code used to fill the cells of the matrix is also presented. Cells marked with "–" mean that the method does not take in consideration the corresponding aspect of knowledge reuse.

6 Related works

To the best of our knowledge, no research exists in the literature to review in a systematic way the issue of knowledge reuse in security requirements engineering. One worth mentioning work though is that of Chernak [109] who conducted an online survey on requirements reuse in 2010. His survey reports that 80 % of participants find that reuse is important and brings benefits. Yoshioka et al. [110] presented a survey limited to security patterns. This is interesting, but the other forms of reuse are neglected, whereas they were taken in consideration in this paper. In a previous work [42], we presented a survey on the use of ontologies in SRE. This work was the start of our research project; it has been extended to other forms of knowledge reuse in SRE in the current paper.

Devanbu et al. [111] is one of the old references that presented a "brief" survey on security models and requirements. Recently, some publications were dedicated to security requirements engineering [27, 112, 113, 114, 115]. Iankoulova et al. [116] propose a systematic review about security requirements in the cloud computing.

However, none of these existing reviews tackled the specific issue of "knowledge reuse." In addition to the presented framework and the SMAP, this paper updates the existing surveys by the latest methods that appeared very recently in the literature.

7 Threats to validity

Like with empirical researches, there are threats to the validity. In the following, some threats that might compromise our results are cited:

7.1 Search engines used in the SMAP (external validity)

All retrieved results rely on the functionality and precision of the search engines of the used digital libraries. Unfortunately, many search engines of computer science digital libraries turned out to be unreliable. Moreover, the results were based on digital libraries for which our institution has subscription to. Unfortunately, we were not able to explore a system like SCOPUS, which is known to be particularly useful because it indexes publications from a large number of publishers.

7.2 Selected sources (construction validity)

In this research, the SMAP was more focused on publications' sources related to the security requirements engineering field than on those related to the knowledge engineering field. This makes the results subject to discussion and comparison with other SMAPs' results that might address the subject in the other way around, "security requirements in knowledge engineering" for example. Moreover, being researchers in the area of requirements engineering and information systems, there is a risk that we may have been biased by our experience and collaborations in the selection and the analysis despite our effort to avoid it. For example, some selected studies of the mapping involved previously the authors or their colleagues. In particular, the papers presenting the method RITA [52, 53] (that included previous researches of one of the authors) were intentionally added to the selected papers. The paper presenting security ontologies [42] cited in the related works section was part of previous researches in the same research project by three of the authors.

The primary search (screening) that was based mainly on title, keywords, and a succinct read of the abstract might have missed relevant papers related to the topic. Some reuse forms like "templates" or "taxonomies" that were discovered through the study were not initially considered in the list of keywords. Moreover, the decision to read or not to read much more than the abstract (for the purpose of selection) strongly depends on the subjective feeling of the authors.

There is another threat related to the number of years that we mention here: The main searches were based on a defined interval of years. The goal of covering a big interval (2000–2013) turned to be ambitious and difficult to manage. There was a need to restrict the number of papers beyond the selected criteria just to make the process manageable and better reported; this might also induce some bias on the final results.

While executing the research protocol, selecting sources is not an easy and straightforward task,in particular, the choice of quality/selection criteria. For example, quality criterion Q1 (publications containing several unsupported claims or frequently referring to existing work without providing a citation were excluded) may lead to controversial opinions. It depends on subjective judgments by the

Table 3 Summary of the systematic mapping study

	Methods (re)using patterns			Methods (re)using taxonomies or ontologies									Methods (re)using templates or profiles	
	KAOS [11] [31]	Secure Troposos [38]	Okubo et al.'s method [41]	Secure i* [89]	GBRAM [44]	Secure Troposos—Si* [51]	RITA [53]	Daramola et al.'s method [55]	Dritsas et al. [58]	Lasheras et al. [59].	Salini et al. [62]	Chikh et al. [63]	Zuccato et al.s method [65]	Firesmith [10]
Form of Representation	P	P	P	–	Tax	O	O	O & T	P, O	O	O	O	Pr	T
Reusable Knowledge	T, C	C, A	A, T, C	–	GV	G, C, A	T, SR	T, SR	C, A, V, T	A, T, C, V	O, T, SR, A, V	T, O, V, A, SR	–	SR
Technique	FR	G	P	–	–	FR	G	Q	Q	Q	Q	–	P	P
Automation (reuse tool)	N	N	N	–	N	N	Y	Y	N	N	N	–	N	N

	Methods (re)using catalogs or generic models						Methods (re)using mixed forms of security knowledge		Methods not (re)using security knowledge							
	Misuse cases [69]	Abuse frames [70]	Security use cases [7]	Saeki and Kaiya's method [74]	SIREN [61]	Secure Troposos [78]	SREP [86]	SQUARE [83]	ISSRM [1]	MSRA[86]	Morda [99]	CRAC++ [101]	CORAS [95]	UMLSec [91]	SREF [104]	Secure UML [90]
Form of Representation	C&GM	GM	C&GM	C&GM	C	C&GM	M	M	–	–	–	–	–	–	–	–
Reusable Knowledge	T, S, R	T, A, V	SR	T, O, C	SR	G, T, C	T, SR	T, SR	–	–	–	–	–	–	–	–
Technique	G	P	G	FR	P	G	–	–	–	–	–	–	–	–	–	–
Automation (reuse tool)	N	N	N	N	N	Y	N	Y	–	–	–	–	–	–	–	–

Form of representation: *P* pattern, *Tax* taxonomy, *O* ontology, *C* catalog, *GM* generic model, *T* template, *Pr* profile, *M* mixed, – null

Reusable knowledge: *T* threats, *C* countermeasures, *A* asset, *O* organization, *G* goal, *V* vulnerabilities, *SR* security requirements, – null

Technique: *FR* formal rules, *G* guidelines, *P* process, *Q* queries, – null

Automation: *N* no, *Y* yes, – null

reviewer, which can only be reduced through feedback from peers. The categorization choices (the map) are another point of discussion. Within the application of the same research protocol, other researchers may decide on a different categorization of the findings.

7.3 Results (Conclusion validity)

The results of the SMAP are useful and interesting; however, these conclusions are based on sources retrieved in conferences, journals, academic, and some industrial reports. It would have been interesting to compare these results with others based on online surveys where real world practitioners are asked about their practices and opinions on security requirements reuse. In addition to that, although there was a careful analysis of the available literature resulting in the presented framework, researches may find that some criteria may have been neglected. Another threat to validity is related to searching exclusively in English writing sources, although it is the largely used language by researchers, but one should pay attention that there are many active communities in other countries who may propose interesting researches related to the topic.

8 Conclusion

Over 30 methods to support SRE engineering were reviewed in this paper. One can safely say that we are now far away from the first generation of "checklist"-based methods as presented by Baskerville [117] in the early nineties. A significant number of publications in the requirements engineering community illustrate the steady interest in security requirements engineering during the last two decades. The area of security knowledge reuse is still emerging. One single mapping study can never be able to cover all aspects of existing contributions. Each one can tackle a single aspect. The richness of the literature allows us to deduce that SRE engineering thus embraces a consequent body of knowledge. The temporal range of the papers considered in our SMAP moreover confirms that this knowledge becomes sufficiently generic to be reused in a systematic way.

This paper presented the details and results of a systematic mapping study conducted to get an extensive overview of existing research on knowledge reuse within SRE. The review provides an overview of important existing approaches and tools. It also proposes a replicable and extensible framework for comparing SRE methodologies. More than 30 approaches covering 13 years of SRE practice were analyzed. Our iterative refinement resulted in a final set of five main types of knowledge forms of representation that were (re)used by SRE approaches: (1) security patterns; (2) taxonomies and ontologies; (3) templates and profiles; (4) catalogs and generic models; and (5) mixed. For each form of representation, more details were provided about the related SRE approach to it, its (re)use, and the tool support provided. A framework to compare and analyze knowledge reuse in SRE was also defined.

The main goal of our SMAP was to provide a good reference to researchers and Ph.D. students to get a clear map on knowledge reuse across SRE and find answers to the different questions on this topic. This is distinguishable comparing to other reviews that compared some existing SRE methods but did not target the knowledge reuse-related questions.

This SMAP can be useful to practitioners (requirements engineers, security officers, security engineers, etc.) who are interested to know what is going in research in terms of SRE. The SMAP provides to practitioners various SRE methods altogether with different knowledge reuse forms (ontologies, patterns, profiles, models, catalogs…). The results of the SMAP can be particularly useful to security architects because they reuse knowledge at corporate level and their responsibilities include to leverage knowledge reuse. For any given set of requirements, an architect can and should typically identify and evaluate multiple different architectures and architectural mechanisms before selecting what he or she thinks will be the optimum way of fulfilling the requirements. Thus, there are often many ways for an architecture or security team to address a specific kind of security requirement. Knowing the different methods can make their job easier. These results will also be useful to beginners in requirement engineering as an aid for training in identifying, analyzing, specifying, and managing security requirements. Requirements teams often do not include subject matter experts in security [10]. Such a body of knowledge can be made available to this intended audience, even if this requires further research and development to make it available in a convenient way.

Our SMAP may help researchers to evaluate both the state of the art in SRE and the open issues due to the limitations in SRE knowledge representation. A further research avenue, for example, could be to explore new knowledge representation models and evaluate how they could enrich the SRE knowledge elicitation process and, consequently, this knowledge reuse. Research on ontologies mention that current ontology languages are limited and that there is a need for semantically richer knowledge models.

The SMAP raises new questions that both research and industrial communities may face:

At the industrial level, the question arises about the real practices of industrials on knowledge reuse during security

requirements elicitations and analyses. What are the good and the bad practices that experts and security consultants may suggest? A survey should be a next step to find answers to these questions.

Moreover, given all these propositions that appeared during the last two decades, what is their maturity (scalability, efficiency) for use in real-life industrial environments? Validation studies are still not sufficient. Such validation process requires a cross-disciplinary work. Moreover, the lack of automated support and the fact that many of the SRE methods rely on reusable knowledge that is not standard remain as issues. The deficiency in automation support may suggest that companies (IT vendors and software producers) still have not invested enough in this domain. This leads to the question about how the automation can be made part of existing security technologies that exist already in companies?

On a research level, many issues may be elicited. Why most of risk-based approaches do not incorporate knowledge reuse? Why is there a lack in automated support that handles knowledge reuse for the different SRE methods? One may claim that conceptual methods are often created as part of Ph.D. researches where automation is not required as part of the dissertation process. However, the research community should be aware of this and should refocus from method creation to automation and then to evaluations to reach a better assessment of the research contributions.

And, even further in the future, can we imagine a collaborative work between researchers and practitioners for a generalization and unification of all these efforts (like in UML), so that their exploitation in practice and even in academic teaching institutions becomes easier?

Appendix: Systematic mapping study: retrieved publications

Table 4 (in the following) presents the searches conducted and the list of all publications retrieved in our systematic mapping study. For each retrieved paper, the following information is provided: name of the first author, title of the paper, year of publication, and digital library/resource. For each category, and for each conference/workshop, the table gives the number of papers found followed by the number of selected papers (using our selection criteria). Finally, the

Table 4 Table of all retrieved papers

First author	Title	Pub. year	Digital library/resource
Books & Book Chapters, Ph.d. Found = 18, Selected = 10			
Mayer, N.	**Model-based management of information system security risk**	**2012**	**Amazon**
Lund, M. S.	*The CORAS tool*	*2011*	*SpringerLink*
Hull, E.	*Requirements engineering*	*2011*	*GoogleBooks*
Yu, E.	*Modeling strategic relationships for process reengineering*	*2011*	*GoogleBooks*
Lopez, J.	**Analysis of security threats, requirements, technologies, and standards in wireless sensor networks**	**2009**	**Foundations of Security Analysis and Design**
Massacci, F.	**An ontology for secure socio-technical systems**	**2007**	**Handbook of Ontologies for Business Interaction.**
Lamsweerde, A.	**Engineering requirements for system reliability and security**	**2007**	**IOS press ebooks**
Giorgini, P.	**Security and trust requirements engineering**	**2005**	**Foundations of Security Analysis and Design**
Jürjens, J.	**Secure systems development with UML**	**2005**	**Amazon**
Ivankina, E.	**An approach to guide requirement elicitation by analyzing the causes and consequences of threats**	**2005**	**Information Modeling and Knowledge Bases**
Kruchten, P.	*The rational unified process: an introduction*	*2004*	*Amazon*
Jackson, M. J	**Problem frames: analyzing & structuring software development problems**	**2001**	**Amazon**
Yu, E.	**Modeling trust for system design using the i* strategic actors framework**	**2001**	**SpringerLink**
Antón, A	**Strategies for developing policies and requirements for secure electronic commerce systems**	**2000**	**SpringerLink**
Jacobson, I.	*The unified software development process*	*1999*	*Amazon*
Kotonya, G.	*Requirements engineering: processes and techniques*	*1998*	*Amazon*
Jackson, M. J.	*Software requirements & specifications: a lexicon of practice, principles, and prejudices*	*1995*	*Amazon*

Table 4 continued

First author	Title	Pub. year	Digital library/resource
Abiteboul, S.	*Foundations of databases*	*1995*	*Amazon*
Journals, Found = 31, Selected = 20			
Computer, Found = 1, Selected = 1			
Nuseibeh, B.	**Securing the skies: in requirements we trust**	**2009**	**IEEE Computer society**
Journal of Electronic Security and Digital Forensics, Found = 1, Selected = 0			
Ivan, F.	*Integrating security and usability into the requirements and design process*	*2007*	*ACM*
Journal of Software Engineering and Knowledge Engineering, Found = 3, Selected = 2			
Mouratidis, H.	**Modeling secure systems using an agent-oriented approach and security patterns**	**2006**	**Google scholar**
Mouratidis, H.	**Secure tropos: a security-oriented extension of the tropos methodology**	**2006**	**Google scholar**
Bauer, B	*Agent UML: a formalism for specifying multiagent software systems*	*2000*	*Citeseerx*
Electronic Journal for E-Commerce Tools and Applications. Found = 1, Selected = 1			
Dritsas, S.	**A knowledge-based approach to security requirements for e-health applications**	**2006**	www.ejeta.org
Autonomous Agents and Multi-Agent Systems, Found = 1, Selected = 1			
Bresciani, P.	**Tropos: an agent-oriented software development methodology**	**2004**	**SpringerLink**
Military Operations Research, Found = 1, Selected = 1			
Buckshaw, D.	**Mission-oriented risk and design analysis of critical information systems**	**2005**	**Ingentaconnect**
Security & Privacy, IEEE, Found = 1, Selected = 1			
Evans, S.	**Risk-based systems security engineering: stopping attacks with intention**	**2004**	**IEEExPlore**
Requirements Engineering Journal, Found = 4, Selected = 3			
Fabian, B.	*A comparison of security requirements engineering methods. Requirements Engineering*	*2010*	*SpringerLink*
Sindre, G.	**Eliciting security requirements with misuse cases**	**2005**	**ACM**
Antón, A	**A requirements taxonomy for reducing Web site privacy vulnerabilities**	**2004**	**SpringerLink**
Toval, A.	**Requirements Reuse for Improving Information Systems Security: a practitioner's approach**	**2001**	**Citeseerx**
Journal of Object Technology, Found = 2, Selected = 2			
Firesmith, D.	**Specifying reusable security requirements**	**2004**	www.jot.fm
Firesmith, D.	**Security use cases**	**2003**	www.jot.fm
Computers & Security, Found = 1, Selected = 0			
Gritzalis, D.	*Principles and requirements for a secure e-voting system*	*2002*	*Sciencedirect*
Software Engineering, IEEE Transactions on. Found = 3, Selected = 1			
Breaux, T.	*Analyzing regulatory rules for privacy and security requirements*	*2008*	*ACM*
Haley, C.B.	**Security requirements engineering: a framework for representation and analysis**	**2008**	**IEEExPlore**
Rolland, C.	*Guiding goal modeling using scenarios*	*1998*	*IEEExPlore*
Computer Standards & Interfaces. Found = 3, Selected = 2			
Mellado, D.	**A common criteria-based security requirements engineering process for the development of secure information systems**	**2007**	**Sciencedirect**
Massacci, F.	**Using a security requirements engineering methodology in practice: The compliance with the Italian data protection legislation**	**2005**	**Sciencedirect**
Bhavani, T.	**Security standards for the semantic web**	**2005**	**Sciencedirect**
Computer Communications. Found = 1, Selected = 0			
Lambrinoudakis, C.	*Security requirements for e-government services: a methodological approach for developing a common PKI-based security policy*	*2003*	*Sciencedirect*
International Journal of Computer Applications. Found = 1, Selected = 1			
Salini, P.	A knowledge-oriented approach to security requirements for an e-voting system	2012	www.ijcaonline.org
Informatical journal. Found = 1, Selected = 1			

Table 4 continued

First author	Title	Pub. year	Digital library/resource
Susi, A.	**The tropos metamodel and its use**	**2005**	http://www.troposproject.org
International Journal of Information Security. Found = 1, Selected = 1			
Giorgini, P.	**Requirements engineering for trust management: model, methodology, and reasoning**	**2006**	**IEEExPlore**
Journal of systems and software. Found = 1, Selected = 1			
Mouratidis H.	**A framework to support selection of cloud providers based on security and privacy requirements**	**2013**	**Sciencedirect**
Journal of Research and Practice in Information Technology. Found = 1, Selected = 1			
Lasheras, J.	**Modeling reusable security requirements based on an ontology framework**	**2009**	**Google scholar**
Information and Software Technology. Found = 1, Selected = 0			
Maamar, Z.	*Toward an ontology-based approach for specifying and securing web services*	*2006*	*Sciencedirect*
Journal of Autonomous Agents and Multi-Agent Systems. Found = 1, Selected = 0			
Kaga, L.	*Modeling conversation policies using permissions and obligations*	*2005*	*ACM*
Engineering Applications of Artificial Intelligence. Found = 1, Selected = 0			
Tan, J. J.	*Dynamic security reconfiguration for the semantic web*	*2004*	*Sciencedirect*
Conferences Found = 70, Selected = 39			
ARES. Found = 12, selected = 6			
Beckers, K.	*Comparing Privacy Requirements Engineering Approaches*	*2012*	*IEEExPlore*
Beckers, K	*Using security requirements engineering approaches to support ISO 27001 information security management systems development and documentation*	*2012*	*IEEExPlore*
Karpati, P.	*Characterizing and analyzing security requirements modeling initiatives*	*2011*	*IEEExPlore*
Kárpáti, P.	*Experimental comparison of misuse case maps with misuse cases and system architecture diagrams for eliciting security vulnerabilities and mitigations*	*2011*	*IEEExPlore*
Okubo, T.	**Effective security impact analysis with patterns for software enhancement**	**2011**	**IEEExPlore**
Zuccato, A.	**Service security requirement profiles for telecom: How software engineers may tackle security?**	**2011**	**IEEExPlore**
Langer, L.	*A taxonomy refining the security requirements for electronic voting: analyzing helios as a proof of concept*	*2010*	*IEEE Computer society*
Schmidt, H.	**Threat and risk analysis during early security requirements engineering**	**2010**	**IEEExPlore**
Hatebur, D.	**A pattern system for security requirements engineering**	**2007**	**IEEExPlore**
Asnar, Y.	**From trust to dependability through risk analysis**	**2007**	**IEEExPlore**
Mellado, D.	*A comparison of the common criteria with proposals of information systems security requirements*	*2006*	*IEEExPlore*
Giorgini, P.	**ST-tool: a CASE tool for security requirements engineering**	**2005**	**IEEExPlore**
AINA, Found = 1, Selected = 1			
Tsoumas, B.	**Toward an ontology-based security management**	**2006**	**IEEExPlore**
CAiSE, Found = 2, Selected = 2			
Paja, E.	**Modeling security requirements in socio-technical systems with STS-tool**	**2012**	**Google scholar**
Mouratidis, H.	**Integrating security and systems engineering: toward the modeling of secure information systems**	**2003**	**Citeseerx**
COMPSAC; Found = 1, Selected = 0			
Elahi, G.	*Security requirements engineering in the wild: a survey of common practices*	*2011*	*IEEExPlore*
CSEE&T. Found = 1, Selected = 1			
Mead, N. R	**Security requirements engineering for software systems: case studies in support of software engineering education**	**2006**	**IEEExPlore**
ETRICS, Found = 1, Selected = 1			
Hatebur, D.	**Security engineering using problem frames**	**2006**	**SpringerLink**
EEE. Found = 1, Selected = 1			
Marti, R.	**Security specification and implementation for mobile e-health services**	**2004**	**IEEExPlore**

Table 4 continued

First author	Title	Pub. year	Digital library/resource
ENASE. Found = 1, Selected = 0			
Semmak, F.	*Extended Kaos to support variability for goal-oriented requirements reuse*	*2010*	*SpringerLink*
FIRA—STA. Found = 1, Selected = 1			
Chikh, A.	**An ontology-based information security requirements engineering framework**	**2011**	**SpringerLink**
HICSS. Found = 1, Selected = 0			
Goluch, G.	*Integration of an ontological information security concept in risk -aware business process management*	*2008*	*IEEExPlore*
ICSE. Found = 3, Selected = 3			
Best, B.,	**Model-based security engineering of distributed information systems using UMLsec**	**2007**	**IEEExPlore**
Firesmith, D.	**Engineering safety and security-related requirements for software-intensive systems**	**2007**	**IEEExPlore**
Van Lamsweed	**Elaborating security requirements by construction of intentional anti-models**	**2004**	**IEEExPlore**
ICSOC. Found = 1, Selected = 0			
Deubler, M.	*Sound development of secure service-based systems*	*2004*	*Citeseer*
ICICS. Found = 1, Selected = 1			
Jensen, J.	Experimental threat model reuse with misuse case diagrams	2010	SpringerLink
IFIP TC9/WG9.6, Found = 2, Selected = 1			
Tsoumas, B.	Security by ontology; A knowledge-centric approach	2006	SpringerLink
Rannenberg, K.	*Recent development in information technology security evaluation: the need for evaluation criteria for multilateral security*	*1993*	*ACM*
iTrust. Found = 1, Selected = 1			
Vraalsen, F.	The CORAS tool for security risk analysis	2005	SpringerLink
MoDELS, Found = 2, Selected = 2			
Saeki, M.	**Security requirements elicitation using method weaving and common criteria**	**2009**	**SpringerLink**
Hogganvik, I.	**A graphical approach to risk identification, motivated by empirical investigations.**	**2006**	**SpringerLink**
NIK. Found = 1, Selected = 1			
Sindre, G.	Capturing security requirements through misuse cases.	2001	Google scholar
RE. Found = 24, Selected = 6			
Morali, A.	**Risk-based confidentiality requirements specification for outsourced IT systems**	**2012**	**IEEE Computer society**
Paja, E.	**STS-tool: Socio-technical Security Requirements through social commitments**	**2012**	**IEEExPlore**
Supakkul, S.	*An NFR Pattern Approach to Dealing with NFRs*	*2010*	*IEEExPlore*
Eunsuk, K.	*Dependability arguments with trusted bases*	*2010*	*IEEExPlore*
Ameller, D.	*Dealing with non-functional requirements in model-driven development*	*2010*	*IEEExPlore*
Hill, J.	*Creating safety requirements traceability for assuring and recertifying legacy safety-critical systems*	*2010*	*IEEExPlore*
Xiping, S.	*Experiences in developing quantifiable NFRs for the service-oriented software platform*	*2009*	*IEEExPlore*
Teng, L.	*AVT vector: a quantitative security requirements evaluation approach based on assets, vulnerabilities, and trustworthiness of environment*	*2009*	*IEEExPlore*
Jureta, I. J.	*Revisiting the core ontology and problem in requirements engineering*	*2008*	*IEEExPlore*
David, C.	*Balancing security requirements and emotional requirements in video games*	*2008*	*IEEExPlore*
Pichler, M.	*Agile requirements engineering for a social insurance for occupational risks organization: a case study*	*2006*	*IEEExPlore*

Table 4 continued

First author	Title	Pub. year	Digital library/resource
Juan, P. C.	*Managing non-technical requirements in cots components selection*	*2006*	*IEEExPlore*
Gonzalez-Baixauli, B.	*Eliciting non-functional requirements interactions using the personal construct theory*	*2006*	*IEEExPlore*
Chisan, J.	*Exploring the role of requirements engineering in improving risk management*	*2005*	*IEEExPlore*
Cohene, T.	*Contextual risk analysis for interview design*	*2005*	*IEEExPlore*
Kaiya, H.	*Identifying stakeholders and their preferences about NFR by comparing use case diagrams of several existing systems*	*2004*	*IEEExPlore*
Haley, C. B.	*The effect of trust assumptions on the elaboration of security requirements*	*2004*	*IEEExPlore*
Lin, L.	**Using abuse frames to bound the scope of security problems**	**2004**	**ACM**
Lin, L.	**Introducing abuse frames for analyzing security requirements**	**2003**	**Citeseerx**
Liu, L.	**Security and privacy requirements analysis within a social setting.**	**2003**	**IEEExPlore**
Steve, L.	*The journey toward secure systems: Achieving Assurance*	*2003*	*IEEExPlore*
Ian, A.	*Initial industrial experience of misuse cases in trade-off analysis*	*2002*	*IEEExPlore*
Wojtek, K.	*Requirements, architectures, and risks*	*2002*	*IEEE Computer society*
Gene, S.	*The hidden meta-requirements of security and privacy*	*2001*	*IEEExPlore*
REFSQ. Found = 3, Selected = 3			
He, Q.	**A framework for modeling privacy requirements in role engineering**	**2003**	http://www4.ncsu.edu
Sindre, G.	**A reuse-based approach to determining security requirements**	**2003**	**Citeseerx**
Sindre, G.	**Templates for misuse case description**	**2001**	**Citeseerx**
SAFECOMP. Found = 1, Selected = 1			
Grünbauer, J.	*Modeling and verification of layered security protocols: a bank application*	*2003*	*SpringerLink*
SKG. Found = 1, Selected = 0			
Vorobiev, A.	*Security attack ontology for web services*	*2006*	*IEEExPlore*
Sicherheit. Found = 1, Selected = 1			
Gürses, S. F	**Contextualizing security goals: a method for multilateral security requirements elicitation**	**2006**	**DBLP**
SIN. Found = 1, Selected = 0			
Parkin, S. E.	*An information security ontology incorporating human-behavioral implications*	*2009*	*ACM*
TrustBus, Found = 1, Selected = 1			
Pavlidis, M.	**Trustworthy selection of cloud providers based on security and privacy requirements: justifying trust assumptions**	**2013**	**IEEE Computer society**
TRUST. Found = 1, Selected = 1			
Vrakas, N.	*Privacy requirements engineering for trustworthy e-government services*	*2010*	*SpringerLink*
TOOLS PACIFIC. Found = 1, Selected = 1			
Sindre, G.	**Eliciting security requirements by misuse cases**	**2000**	**IEEExPlore**
UML. Found = 2, Selected = 2			
Jürjens, J.	**Automated verification of UMLsec models for security requirements**	**2004**	**Citeseerx**
Lodderstedt, T.	**SecureUML: A UML-based modeling language for model-driven security**	**2002**	**ACM**
WWW. Found = 1, Selected = 1			
Martí, R.	*Security in a wireless mobile health care system.*	*2003*	*Google scholar*
Workshops, Found = 21, Selecetd = 11			
ASIACCS. Found = 1, Selected = 0			
Fenz, S.	*Formalizing information security knowledge*	*2009*	*Citeseerx*
CAiSE workshops. Found = 1, Selected = 0			
Massacci, F.	*An extended ontology for security requirements*	*2011*	*SpringerLink*
DEXA. Found = 1, Selected = 1			

Table 4 continued

First author	Title	Pub. year	Digital library/resource
Hatebur, D.	**A security engineering process based on patterns**	**2007**	**IEEExPlore**
ESORICS. Found = 1, Selected = 1			
Mellado, D.	**Applying a security requirements engineering process**	**2006**	**SpringerLink**
EDOCW. Found = 1, Selected = 0			
Naufel do Amaral, F.	*An ontology-based approach to the formalization of information security policies*	*2006*	*IEEExPlore*
EWSA. Found = 1, Selected = 0			
Schmidt, H.	*Preserving software quality characteristics from requirements analysis to architectural design*	*2006*	*IEEExPlore*
MARK. Found = 1, Selected = 1			
Salinesi, C.	Using the RITA threats ontology to guide requirements elicitation: an empirical experiment in the banking sector	2008	IEEExPlore
NSPW. Found = 1, Selected = 0			
Raskin, V.	*Ontology in information security: a useful theoretical foundation and methodological tool*	*2001*	*ACM*
OTM. Found = 1, Selected = 1			
Daramola, O.	**Ontology-based support for security requirements specification process**	**2012**	**Springer**
RePa. Found = 1, Selected = 1			
Daramola, O.	**Pattern-based security requirements specification using ontologies and boilerplates**	**2012**	**IEEExPlore**
RISI. Found = 1, Selected = 0			
Beckers, K.	*An integrated method for pattern-based elicitation of legal requirements applied to a cloud computing example*	*2012*	*DBLP*
RHAS. Found = 2, Selected = 1			
Firesmith, D.	A taxonomy of security-related requirements	2005	Citeseerx
Lee, S. W.	*Engineering dependability requirements for software-intensive systems through the definition of a common language*	*2005*	*Citeseerx*
SecSE. Found = 1, Selected = 0			
Seeger, M. M.	*A comparative study of software security pattern classifications*	*2012*	*DBLP*
SESS. Found = 3, Selected = 2			
Lee, S. W.	*Building problem domain ontology from security requirements in regulatory documents*	*2006*	*ACM*
Haley, C. B.	**A framework for security requirements engineering.**	**2006**	open.ac.uku
Mead, N. R.	**Security quality requirements engineering (SQUARE) methodology**	**2005**	**ACM**
SAC. Found = 1, Selected = 1			
Jürjens, J.	**Using UMLsec and goal trees for secure systems development**	**2002**	**ACM**
Spattern. Found = 1, Selected = 1			
Fernandez, E. B.	*Measuring the level of security introduced by security patterns*	*2010*	*IEEExPlore*
UKDU Workshop. Found = 1, Selected = = 1			
Gürses, S.	Multilateral security requirements analysis for preserving privacy in ubiquitous environments	2006	Citeseer
WSCS. Found = 1, Selected = 0			
Yang, Y.	*Toward semantic requirement engineering*	*2008*	*IEEExPlore*
Reports, Found = 18, Selected = 15			
Wenzel, S.	**Approach for adaptive security monitor generation**	**2012**	http://www.securechange.eu
MAGERIT v.3	**Metodología de Análisis y Gestión de Riesgos de los Sistemas de Información**	**2012**	http://administracionelectronica.gob.es
Travis, C.	**Security requirements reusability and the square methodology**	**2010**	sei.cmu.edu
Morali, A.	**CRAC: Confidentiality risk analysis and IT-architecture comparison of business networks**	**2009**	http://doc.utwente.nl/

Table 4 continued

First author	Title	Pub. year	Digital library/resource
Mead, N. R	**Incorporating security quality requirements engineering (SQUARE) into standard life-cycle models**	**2008**	**Google scholar**
Elahi, G.	**A goal-oriented approach for modeling and analyzing security trade-Offs**	**2007**	**Citeseerx**
Dahl, H.	**Structured semantics for the CORAS security risk modeling language**	**2007**	coras.sourceforge.net/
Hermoye, L. A.	**Attack patterns for security requirements engineering**	**2006**	**Google scholar**
Mylopoulos, J.	**Risk modeling and reasoning in goal models**	**2006**	http://eprints.biblio.unitn.it
Hermoye, L. A.	**A reuse-based approach to security requirements engineering**	**2006**	users.ece.utexas.edu
Massacci, F.	**Detecting conflicts between functional and security requirements with secure tropos**	**2006**	**Citeseerx**
Araujo, R.	**Design authorization systems using secure UML**	**2005**	**Google**
Lin, L.-C.	**Analyzing security threats and vulnerabilities using abuse frames**	**2003**	**Google scholar**
Firesmith	**Common concepts underlying safety, security, and survivability engineering**	**2003**	sei.cmu.edu
MD, N. S. A. S. S. F. G. G. M	**Common criteria for information technology security evaluation**	**2002**	**Google scholar**
Lin, L.	**Analyzing security requirements as relationships among strategic actors**	**2002**	**Citeseerx**
Schmidt, H.	*UML revision taskforce*	*2001*	*Citeseerx*
ECMA-271	*Extended commercially oriented functionality class for security evaluation*	*1999*	*Google*

bold emphasis refers to papers selected; the italic emphasis refers to papers not selected in the SMAP.

References

1. Mayer N (2012) Model-based management of information system security risk. Presses universitaires de Namur, Namur, Belgieum
2. Giorgini P, Massacci F, Zannone N (2005) Security and trust requirements engineering. In: Aldini A, Gorrieri R, Martinelli F (eds) Foundations of security analysis and design III. Springer, Berlin, pp 237–272
3. Liu L, Yu E, Mylopoulos J (2002) Analyzing security requirements as relationships among strategic actors. In: Proceedings of the 2nd symposium on requirements engineering for information security
4. Mouratidis H (2006) Analysing security requirements of information systems using tropos, January-2006. (Online). http://roar.uel.ac.uk/409/. Consulted 17 Nov 2012
5. Van Lamsweerde A (2004) Elaborating security requirements by construction of intentional anti-models. In: Proceedings of 26th international conference on software engineering, 2004. ICSE 2004, pp 148–157
6. Sindre G, Opdahl AL (2005) Eliciting security requirements with misuse cases. Requir Eng 10(1):34–44
7. Firesmith DG (2003) Security use cases. J Object Technol 2(3):53–64
8. Lodderstedt D, Basin J Doser (2002) SecureUML: a UML-based modeling language for model-driven security. In: Jézéquel J-M, Hussmann H, Cook S (eds) ≪UML ≫ 2002—The Unified Modeling Language. Springer, Berlin, pp 426–441
9. Jürjens J (2002) Using UMLsec and goal trees for secure systems development. In: Proceedings of the 2002 ACM symposium on Applied computing, New York, 2002, pp 1026–1030
10. Firesmith D (2004) Specifying reusable security requirements. J Object Technol 3(1):61–75
11. Hermoye LA, van Lamsweerde A, Perry DE (2014) A reuse-based approach to security requirements engineering. (Online). http://users.ece.utexas.edu/∼perry/work/papers/060908-LH-reuse.pdf. Consulted 17 Dec 2014
12. Jensen J, Tøndel IA, Meland PH (2010) Experimental threat model reuse with misuse case diagrams. In: Soriano M, Qing S, López J (eds) Information and communications security. Springer, Berlin, pp 355–366
13. Hatebur D, Heisel M, Schmidt H (2007) A pattern system for security requirements engineering. In: The second international conference on availability, reliability and security, 2007. ARES 2007, pp 356–365
14. Heineman GT, Councill WT (2001) Component-based software engineering: putting the pieces together (paperback), 1st edn. Addison-Wesley Professional, Upper Saddle River
15. Frakes WB, Kang K (2005) Software reuse research: status and future. IEEE Trans Softw Eng 31(7):529–536
16. Lam W, McDermid JA, Vickers AJ (1997) Ten steps towards systematic requirements reuse. Requir Eng 2(2):102–113
17. Robertson S, Robertson J (2013) Mastering the requirements process getting requirements right. Addison-Wesley, Upper Saddle River
18. López O, Laguna MA, Peñalvo FJG (2002) Metamodeling for requirements reuse. In: WER, 2002, pp 76–90
19. Walton P, Maiden N (1993) Integrated software reuse: management and techniques. Ashgate Publishing Company Brookfield, VT, USA
20. Mead NR, McGraw G (2005) A portal for software security. IEEE Secur Priv 3(4):75–79
21. Petersen K, Feldt R, Mujtaba S, Mattsson M (2008) Systematic mapping studies in software engineering. In: 12th international conference on evaluation and assessment in software engineering, vol. 17, p 1, 2008
22. Kitchenham B, Charters S (2007) Guidelines for performing systematic literature reviews in software engineering. In: Technical report EBSE-2007-01, 2007

23. Budgen D, Turner M, Brereton P, Kitchenham B (2008) Using mapping studies in software engineering. Proc PPIG 8:195–204
24. Kitchenham BA, Budgen D, Brereton OP (2011) Using mapping studies as the basis for further research–a participant-observer case study. Inform Softw Technol 53(6):638–651
25. Wieringa R, Maiden N, Mead N, Rolland C (2006) Requirements engineering paper classification and evaluation criteria: a proposal and a discussion. Requir Eng 11(1):102–107
26. Dubois É, Heymans P, Mayer N, Matulevičius R (2010) A systematic approach to define the domain of information system security risk management. In: Nurcan S, Salinesi C, Souveyet C, Ralyté J (eds) Intentional perspectives on information systems engineering. Springer, Berlin, pp 289–306
27. Fabian B, Gürses S, Heisel M, Santen T, Schmidt H (2010) A comparison of security requirements engineering methods. Requir Eng 15(1):7–40
28. Elahi G (2009) Security requirements engineering: state of the art and practice and challenges. http://www.cs.utoronto.ca/~gelahi/Depth
29. Mouratidis H, Giorgini P, Schumacher M, Manson M (2003) Security patterns for agent systems. In: Proceedings of the eight european conference on pattern languages of programs (EuroPLoP), Irsee, Germany, 2003
30. van Lamsweerde A (2007) Engineering requirements for system reliability and security. In: Broy JGM, Hoare C (eds) Software system reliability and security, ser. NATO security through science series-D: information and communication security, vol 9. IOS Press, Amsterdam, The Netherlands, pp 196–238
31. Hermoye LA, van Lamsweerde A, Perry DE (2006) Attack patterns for security requirements engineering. September, (Online), 2006. https://hostdb.ece.utexas.edu/~perry/work/papers/060908-LH-threats.pdf
32. Bresciani P, Perini A, Giorgini P, Giunchiglia F, Mylopoulos J (2004) Tropos: an agent-oriented software development methodology. Auton Agents Multi-Agent Syst 8(3):203–236
33. Susi A, Perini A, Mylopoulos J, Giorgini P (2005) The tropos metamodel and its use. Informatica (Slovenia) 29(4):401–408
34. Mouratidis H, Giorgini P (2007) Secure Tropos: a security-oriented extension of the tropos methodology. Int J Softw Eng Knowl Eng 17(2):285–309
35. Mouratidis H, Giorgini P, Manson G (2003) Integrating security and systems engineering: towards the modelling of secure information systems. In: Proceedings of the 15th conference on advanced information systems engineering CAiSE, 2003, pp 63–78
36. Pavlidis M, Mouratidis H, Kalloniatis C, Islam S, Gritzalis S (2013) Trustworthy selection of cloud providers based on security and privacy requirements: justifying trust assumptions. In: Furnell S, Lambrinoudakis C, Lopez J (eds) Trust, privacy, and security in digital business. Springer, Berlin, pp 185–198
37. Paja E, Dalpiaz F, Poggianella M, Roberti P, Giorgini P (2012) STS-Tool: using commitments to specify socio-technical security requirements. In: Castano S, Vassiliadis P, Lakshmanan LV, Lee ML (eds) Advances in conceptual modeling. Springer, Berlin, pp 396–399
38. Mouratidis H, Weiss M, Giorgini P (2006) Modelling secure systems using an agent-oriented approach and security patterns. Int J Software Eng Knowl Eng 16:471–498
39. Alexander C, Ishikawa S, Silverstein M (1977) A pattern language: towns, buildings, construction. Oxford University Press, New York
40. Giorgini P, Massacci F, Mylopoulos J, Zannone N (2005) ST-tool: a CASE tool for security requirements engineering. In: Proceedings of 13th IEEE international conference on requirements engineering, 2005, pp 451–452
41. Okubo T, Kaiya H, Yoshioka N (2011) Effective security impact analysis with patterns for software enhancement. In: 2011 sixth international conference on availability, reliability and security (ARES), 2011, pp 527–534
42. Souag A, Salinesi C, Comyn-Wattiau I (2012) Ontologies for security requirements: a literature survey and classification. In: Advanced information systems engineering workshops lecture notes in business information processing, vol 112, pp 61–69
43. Antón AI, Earp JB (2001) Strategies for developing policies and requirements for secure electronic commerce systems. In: Ghosh AK (ed) E-commerce security and privacy. Kluwer Academic Publishers, Dordrecht, pp 29–46
44. He Q, Anton AI (2003) A framework for modeling privacy requirements in role engineering, international workshop on requirements engineering for software quality (REFSQ 2003), Klagenfurt/Velden, Austria, 16–17 June, 2003
45. Antón AI, Earp JB (2004) A requirements taxonomy for reducing web site privacy vulnerabilities. Requir Eng 9(3):169–185
46. Giorgini P, Massacci F, Mylopoulos J, Zannone N (2006) Requirements engineering for trust management: model, methodology, and reasoning. Int J Inf Secur 5(4):257–274
47. Massacci F, Prest M, Zannone N (2004) Using a security requirements engineering methodology in practice: the compliance with the Italian data protection legislation. University of Trento, Departmental Technical Report, November 2004
48. Massacci F, Zannone N (2008) Detecting conflicts between functional and security requirements with Secure Tropos: John Rusnak and the Allied Irish Bank. In: Yu E, Giorgini P, Maiden N, Mylopoulos J (eds) Social modeling for requirements engineering. MIT Press, Cambridge
49. Asnar Y, Giorgini P, Massacci F, Zannone N (2007) From trust to dependability through risk analysis. In: The second international conference on availability, reliability and security, 2007. ARES 2007. IEEE, pp 19–26
50. Asnar Y, Giorgini P, Mylopoulos J (2006) Risk modelling and reasoning in goal models. University of Trento, Departmental Technical Report, February 2006
51. Massacci F, Mylopoulos J, Zannone N (2007) An ontology for secure socio-technical systems. Handbook of ontologies for business interaction, 2007
52. Ivankina E (2005) An approach to guide requirement elicitation by analysing the causes and consequences of threats. Inform Model Knowl. Bases XVI 121:13
53. Salinesi C, Ivankina E, Angole W (2008) Using the RITA threats ontology to guide requirements elicitation: an empirical experiment in the banking sector. In: Managing requirements knowledge, 2008. MARK'08. First International Workshop on, 2008, pp 11–15
54. Rolland C, Souveyet C, BenAchour C (1998) Guiding goal modeling using scenarios. IEEE Trans Softw Eng 24(12):1055–1071
55. Daramola O, Sindre G, Moser T (2012) Ontology-based support for security requirements specification process. In: Herrero P, Panetto H, Meersman R, Dillon T (eds) On the move to meaningful internet systems: OTM 2012 workshops. Springer, Berlin, pp 194–206
56. Daramola O, Sindre G, Stalhane T (2012) Pattern-based security requirements specification using ontologies and boilerplates. In: 2012 IEEE second international workshop on requirements patterns (RePa), 2012, pp 54–59
57. Hull E (2011) Requirements engineering. Springer, London
58. Dritsas S, Gymnopoulos L, Karyda M, Balopoulos T, Kokolakis S, Lambrinoudakis C, Katsikas S (2006) A knowledge-based approach to security requirements for e-health applications.

Electron J E-Commer Tools Appl, In the Special Issue: Emerging Security Paradigms in the Knowledge Era, 2(1)
59. Velasco JL, Valencia-Garcia R, Fernandez-Breis JT, Toval A (2009) Modelling reusable security requirements based on an ontology framework. J Res Pract Inf Technol 41(2):119
60. PAe - MAGERIT v.3: Metodología de Análisis y Gestión de Riesgos de los Sistemas de Información. (Online). http://administracionelectronica.gob.es/pae_Home/pae_Documentacion/pae_Metodolog/pae_Magerit.html#.UhAEVLxsDR0. Consulted: 17 Aug 2013
61. Toval A, Nicolás J, Moros B, García O (2001) Requirements reuse for improving information systems security: a practitioner's approach. Requir Eng J 6:205–219
62. Salini P, Kanmani S (2012) A knowledge-oriented approach to security requirements for an E-voting system. Int J Comput Appl 49(11):21–25
63. Chikh A, Abulaish M, Nabi SI, Alghathbar K (2011) An Ontology based information security requirements engineering framework. In: Park JJ, Lopez J, Yeo S-S, Shon T, Taniar D (eds) Secure and trust computing, data management and applications. Springer, Berlin, pp 139–146
64. Fenz S, Ekelhart A (2009) Formalizing information security knowledge. (ASIACCS'09), pp 183–194
65. Zuccato A, Daniels N, Jampathom C (2011) Service security requirement profiles for telecom: how software engineers may tackle security. In: 2011 sixth international conference on availability, reliability and security (ARES), 2011, pp 521–526
66. Sindre G, Opdahl AL (2001) Templates for misuse case description. In: Proceedings of the 7th international workshop on requirements engineering, foundation for software quality, REFSQ'2001, 2001, pp 4–5
67. Sindre G, Opdahl AL (2001) Capturing security requirements through misuse cases. NIK 2001, Norsk Informatikkonferanse 2001. http://www.nik.no/2001
68. Alexander I (2002) Initial industrial experience of misuse cases in trade-off analysis. In: Proceedings of IEEE joint international conference on requirements engineering, 2002, pp 61–68
69. Sindre G, Firesmith DG, Opdahl AL (2003) A reuse-based approach to determining security requirements. In Proceedings of 9th international workshop on requirements engineering: foundation for software quality, REFSQ'03, 2003, pp 16–17
70. Lin L, Nuseibeh B, Ince D, Jackson M, Moffett J (2003) Introducing abuse frames for analysing security requirements. In: Proceedings of 11th IEEE International requirements engineering conference, RE'03, 2003, pp 371–372
71. Lin L, Nuseibeh B, Ince D, Jackson M, Moffett J (2003) Analysing security threats and vulnerabilities using abuse frames. ETAPS-04, 2003
72. Lin L, Nuseibeh B, Ince D, Jackson M Using abuse frames to bound the scope of security problems. In: Not Set (ed) 12th IEEE international requirements engineering conference (RE'04). IEEE Computer Society, pp 354–355
73. Jackson MJ (2001) Problem frames: analysing and structuring software development problems. Addison-Wesley/ACM Press, Harlow
74. Saeki M, Kaiya H (2009) Security requirements elicitation using method weaving and common criteria. In: Chaudron MRV (ed) Models in software engineering. Springer, Berlin, pp 185–196
75. N. S. A. S. S. F. G. G. M. MD Common criteria for information technology security evaluation. Department of Defense Public Key Infrastructure and Key Management Infrastructure Token Protection Profile (Medium Robustness, March 2002
76. Common Criteria for Information Technology Security Evaluation. Part 2: Security Functional components. https://www.commoncriteriaportal.org/files/ccfiles/CCPART2V3.1R4.pdf. Consulted: 28 Sept 2013
77. ECMA-271 (1999) Extended commercially oriented functionality class for security evaluation, E-COFC
78. Mouratidis H, Islam S, Kalloniatis C, Gritzalis S (2013) A framework to support selection of cloud providers based on security and privacy requirements. J Syst Softw 86(9): 2276–2293
79. Mead NR, Stehney T (2005) Security quality requirements engineering (SQUARE) methodology. In: Proceedings of the 2005 workshop on Software engineering for secure system building trustworthy applications, New York, 2005, pp 1–7
80. Mead NR, Viswanathan V, Padmanabhan D, Raveendran A (2008) Incorporating security quality requirements engineering (SQUARE) into standard life-cycle models, May 2008
81. Mead NR, Hough ED (2006) Security requirements engineering for software systems: case studies in support of software engineering education. In: Proceedings of 19th conference on software engineering education and training, 2006, pp 149–158
82. Rannenberg K (1993) Recent development in information technology security evaluation-the need for evaluation criteria for multilateral security. In: Security and control of information technology in society, 1993, pp 113–128
83. Christian T (2010) Security requirements reusability and the SQUARE methodology, No. CMU/SEI-2010-TN-027. Carnegie-Mellon University, Software Engineering Institute, Pittsburgh
84. Mellado D, Fernandez-Medina E, Piattini M (2008) Security requirements variability for software product lines. In: Third international conference on availability, reliability and security, 2008. ARES 08, 2008, pp 1413–1420
85. Mellado D, Fernández-Medina E, Piattini M (2006) Applying a security requirements engineering process. In: Gollmann D, Meier J, Sabelfeld A (eds) Computer security—ESORICS 2006. Springer, Berlin, pp 192–206
86. Mellado D, Fernández-Medina E, Piattini M (2007) A common criteria based security requirements engineering process for the development of secure information systems. Comput Stand Interfaces 29(2):244–253
87. Jacobson I, Booch G, Rumbaugh J (1999) The unified software development process. Addison-Wesley, Reading
88. Yu E, Liu L (2001) Modelling trust for system design using the i* strategic actors framework. In: Falcone R, Singh M, Tan Y-H (eds) Trust in cyber-societies. Springer, Berlin, Heidelberg, pp 175–194
89. Liu L, Yu E, Mylopoulos J (2003) Security and privacy requirements analysis within a social setting. In: Proceedings of 11th IEEE international requirements engineering conference, 2003, pp 151–161
90. Araujo R, Gupta S (2005) Design authorization systems using secureUML. In: Foundstone foundstone professional services, 2005, pp 2–16
91. Jürjens J (2005) Secure systems development with UML. Springer, Berlin
92. Jürjens J, Shabalin P (2004) Automated verification of UMLsec models for security requirements. In UML 2004—The Unified Modeling Language, volume 2460 of LNCS, 2004, pp 412–425
93. Best B, Jurjens J, Nuseibeh B (2007) Model-based security engineering of distributed information systems using UMLsec. In: 29th international conference on software engineering, 2007, pp 581–590
94. Wenzel S, Warzecha D, Jurjens J (2012) Approach for adaptive security monitor generation—secureChange. yumpu.com, 31 Jan 2012. (Online). http://www.yumpu.com/en/document/view/8097461/approach-for-adaptive-security-monitor-generation-securechange. Consulted: 26 Aug 2013
95. Dahl HEI, Hogganvik I, Stølen K (2007) Structured semantics for the CORAS security risk modeling language. In: Pre-

proceedings of the 2nd international workshop on interoperability solutions on trust, security, policies and QoS for enhanced enterprise systems (IS-TSPQ'07), pp 79–92
96. Lund MS, Solhaug B, Stølen K (2011) The CORAS tool. In: Model-driven risk analysis. Springer, Berlin, Heidelberg, pp 339–346
97. Vraalsen F, den Braber F, Lund MS, Stølen K (2005) The CORAS tool for security risk analysis. In: Herrmann P, Issarny V, Shiu S (eds) Trust management. Springer, Berlin, pp 402–405
98. Hogganvik I, Stølen K (2006) A graphical approach to risk identification, motivated by empirical investigations. In: Proceedings of the 9th international conference on model driven engineering languages and systems, Berlin, 2006, pp 574–588
99. Evans S, Heinbuch D, Kyle E, Piorkowski J, Wallner J (2004) Risk-based systems security engineering: stopping attacks with intention. IEEE Secur Priv 2(6):59–62
100. Buckshaw DL, Parnell GS, Unkenholz WL, Parks DL, Wallner JM, Saydjari OS (2005) Mission oriented risk and design analysis of critical information systems. Mil Oper Res 10(2):19–38
101. Morali A, Wieringa R (2010) Risk-based confidentiality requirements specification for outsourced IT systems. In: 2010 18th IEEE international requirements engineering conference (RE), 2010, pp 199–208
102. CRAC: Confidentiality risk analysis and IT-architecture comparison of business networks. (Online). http://www.tue.nl/en/publication/ep/p/d/ep-uid/231273/. Consulted: 25 Sept 2013
103. Haley CB, Laney R, Moffett JD, Nuseibeh B (2008) Security requirements engineering: a framework for representation and analysis. IEEE Trans Softw En 34(1):133–153
104. Haley CB, Moffett JD, Laney R, Nuseibeh B (2006) A framework for security requirements engineering. In: Proceedings of the 2006 international workshop on software engineering for secure systems, New York, 2006, pp 35–42
105. Nuseibeh B, Haley CB, Foster C Securing the skies: in requirements we trust. Computer, In: IEEE Computer Society, pp 46–54
106. Gürses S, Berendt B, Santen T (2006) Multilateral security requirements analysis for preserving privacy in ubiquitous environments. In: Proceedings of the UKDU workshop, 2006, pp 51–64
107. Gürses SF, Santen T (2006) Contextualizing security goals: a method for multilateral security requirements elicitation. In: Sicherheit, 2006, vol 6, pp 42–53
108. Souag A, Salinesi C, Mazo R, Comyn-Wattiau I (2015) A security ontology for security requirements elicitation. In: International symposium on engineering secure software and systems, March 4–6, 2015. To appear
109. Chernak Y (2012) Requirements reuse: the state of the practice. In: 2012 IEEE international conference on software science, technology and engineering (SWSTE), 2012, pp 4653
110. Yoshioka N, Washizaki H, Maruyama K (2008) A survey on security patterns. Prog Inf 5(5):35–47
111. Devanbu PT, Stubblebine S (2000) Software engineering for security: a roadmap. In: Proceedings of the conference on the future of software engineering. ACM, pp 227–239
112. Mellado D, Blanco C, Sánchez LE, Fernández-Medina E (2010) A systematic review of security requirements engineering. Comput Stand Interfaces 32(4):153–165
113. Salini P, Kanmani S (2012) Survey and analysis on security requirements engineering. Comput Electron Eng 38(6):1785–1797
114. Tondel IA, Jaatun MG, Meland PH (2008) Security requirements for the rest of us: a survey. Softw IEEE 25(1):20–27
115. Elahi G (2009) Security requirements engineering: state of the art and practice and challenges. http://www.cs.utoronto.ca/~gelahi/DepthPaper.pdf
116. Iankoulova I, Daneva M (2012) Cloud computing security requirements: a systematic review. In: Sixth International conference on research challenges in information science (RCIS). IEEE, 2012, pp 1–7
117. Baskerville Richard (1993) information systems security design methods: implications for information systems development. ACM Comput Surv (CSUR) 25(4):375–414