A View-based Approach for Service-Oriented Security Architecture Specification

Aleksander Dikanski, Sebastian Abeck
Research Group Cooperation & Management (C&M)
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany
{a.dikanski, abeck}@kit.edu

***Abstract*—Developing secure software is still a software engineering challenge because of the complexity of software security. Yet integrating security engineering and software engineering is increasingly important, especially for service-oriented applications, as they are exposed to new security challenges due to their open nature. Current security engineering approaches do not consider existing security architectures, leading to redundant development of security artifacts. Further, present security architecture approaches do not provide relevant information to a security engineering process. Using a service-oriented and security architecture-centric approach for security engineering supports the development of secure service-oriented applications, as existing security solutions can be reused. In this paper a model for service-oriented security architectures is presented, which provides apt information to different consumers, such as security engineering processes and business services, in the form of views to assist the consumers security goals. The architecture model is exemplified by specifying different views of a web service-based security architecture.**

Keywords-security architecture; security engineering; service-orientation; web service, security services.

I. INTRODUCTION

Software engineering is focused on developing required functionality [1], but with increasing software support for business-critical processes, the importance of software security grows, in order to prevent financial loss, damage of reputation or data leakage [2]. Security engineering, i.e., the engineering of software which functions correctly under malicious attacks [1][3], requires the incorporation of security into all development process phases [4]. Yet the intrinsic specifics of security knowledge and the huge amount of complex security standards prevent such integration, leading to post-hoc consideration of security measures [5][6].

Due to the increasing use of networked software services, the focus of security engineering has shifted from software security to application security, i.e., using security products to secure already existing applications, components and services [7]. The paradigm of service-oriented architectures (SOA) established itself as the main architectural concept for today's enterprise information systems, as it allows traditional software systems to be restructured as reusable software services [8][9][10][11]. These software systems are now exposed to a vast quantity of new threats and attacks they were not designed for and need to be protected by external security services [7].

Traditional security engineering approaches focus on eliciting and specifying security requirements and choosing appropriate security measures in order to secure single software systems. Current security engineering approaches for secure SOA applications tend to duplicate these procedures in order to provide security measures for single services. They thereby ignore existing security infrastructures, in which enterprises have invested large sums of money in to protect their existing software assets and which should be reused and restructured to security services according to the SOA paradigm. Additionally, existing security infrastructures provide constraints for security measures, which can be reused in a security engineering approach. On the other hand side, approaches for structuring the security infrastructure of an enterprise into a service-oriented security architectures do not consider this engineering view, instead they focus on the internal architecture and provided services without taking security requirements of service-oriented applications into account.

The contribution of this paper is the specification of a service-oriented security architecture model incorporating different interrelated views for security engineering, security infrastructure integration and security services in order to support the development and operation of secure service-oriented applications. The security engineering view provides development-time information artifacts, such as predefined security policy models and architectural constraints, as well as technology guidelines in order to support the development of secure service-oriented applications. The integration view specifies how existing security components and applications are restructured as security services. The service view specifies how application services can delegate their security requirements to the security services.

The rest of this paper is structured as follows: In Section 2, related approaches of security engineering and security architecture are discussed. In Section 3, our service-oriented security architecture model is presented and each view is discussed. Section 4 contains the specification of a concrete security architecture based on web service technology using our abstract specification, which was developed for a service-oriented navigation system. A conclusion and an outlook on future research directions conclude the body of this paper.

II. Background and Related Work

A. Security Engineering

Software security knowledge reaches from secure code to enterprise wide security products such as firewalls. Each of these areas includes specific terms, standards and technologies that most developers are not familiar with. In order to support the development of secure software, security engineering intends to provide principles, methodologies, and tools for designing, developing, operating, and maintaining secure software systems [1].

Our research revealed only a few approaches for the methodical development of secure software. In [12][13], a general purpose security development process for traditional software is described using security patterns [15][16]. In [14], security problem frames, an adapted version of problem frames [17], are used to specify security requirements and to derive security components. Security engineering approaches for the development of SOA applications can be found in [18] and [6], which are motivated by the complexity of current web service security standards. Reference [18] uses patterns and model-driven software development techniques to automatically generate security standards artifacts. In [6] a process solely for the development of secure web service-based systems is presented. All of the approaches tend to generated new security measures and fail to map these to existing security infrastructures and services, which is a general requirement for SOA applications.

As the development of secure software needs to be embedded into a development method for software functionality, a larger amount of approaches focus on particular phases of security engineering processes. Misuse cases [19][20] and misuse activities [21][22] have been used to describe hostile intentions, attacks, and unwanted behavior in order to elicit security requirements. [23] uses extended problem frames to provide a formal specification of security requirements. Security patterns [15][16] are used to design security measures using best practice solutions. Each of these approaches supports a security engineering process, yet we have found no work, which shows their combined usage. It can also be argued, that these approaches are hard to adapt to the development of secure SOA applications, as it differs from the development of traditional software. Additionally, using each of these approaches leads to an increased complexity for developing secure measures, which matches the main development of the applications functionality.

In order to simplify the specifics of security, an increased interest in reusable security development artifacts can be registered in literature, such as reusable misuse cases, security requirements [24] and security patterns [15][16]. Yet developers still need to be supported in choosing appropriate items from a catalog of reusable security artifacts. A service-oriented approach can be helpful by providing a common set of reusable security artifacts to a development process, which are based on the currently used security measures implemented in a service-oriented security architecture, containing all information of previously developed secure SOA applications and services.

B. Service-Oriented Security Architectures

As there are several approaches for security architectures, we concentrated on the approaches for service-oriented security architectures in our literature review. In [25] the authors describe an event-driven reference architecture for three types of security services namely authentication, access control and identity management service, which communicate through the dispatching of security events. Similar service types are presented in [26], but the authors distinguish a service interface layer, providing security services to business-related services and layers of logical components implementing the security service functionalities. The logic components can be replaced by an existing security product as the authors show in [27]. The authors of [28] specify a technical security architecture for securing message exchange in business.

As each of these approaches focuses on different security measures, i.e., access control and message security, they never provide a complete picture of a security architecture. In neither of the approaches the application requirements, which lead to the specific structure of the architecture, are mentioned. As the services provided by a security architecture are determined by applications' security requirements, the presented architectures exhibit only very generic structures and need to be adapted to concrete security requirements.

The service-oriented integration of security products [29] is only directly considered in the approach of [26][27]. Their service access points only provide security services to service consumers, leaving the service implementation open. Using the service interfaces and event-driven approach for communication between the security services in [25] makes an integration not feasible as the efficient internal communication paths of security products are interrupted.

According to the SOA paradigm, services should provide their functionality through open and standardized interfaces in order to be reusable by different service consumers. Most of the aforementioned approaches rely on web service technology and related standards for service interface description. They thereby mix a technology-oriented view with the abstract description of their reference architectures, making it hard to adapt their approaches to other technology platforms. They also do not provide specific guidelines or conventions on how to apply the used standards, which would support the usage of the complex and flexible web service standards and related security standards.

Each approach has its advantages but neither alone provides enough support for software developers to handle software security complexity. So far, security engineering processes neglect existing and implemented security solutions, leading to the development of redundant security measures. Security architectures do not provide their intrinsic knowledge about implemented security solutions to security engineering processes. By providing such knowledge, software developers are supported in their task of choosing and implementing appropriate security measures.

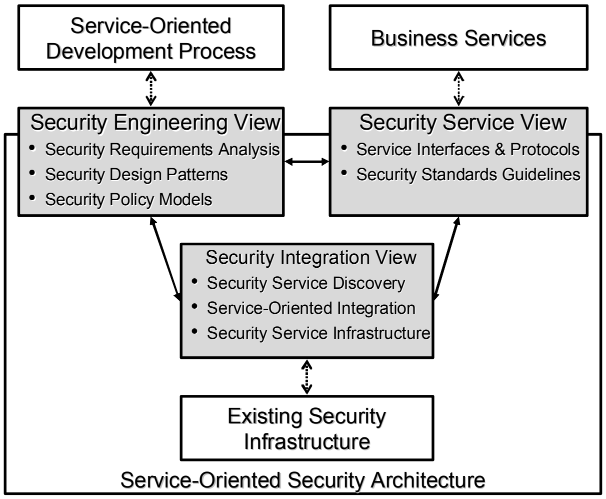
Service-Oriented Development Process
Business Services
Security Engineering View
• Security Requirements Analysis
• Security Design Patterns
• Security Policy Models
Security Service View
• Service Interfaces & Protocols
• Security Standards Guidelines
Security Integration View
• Security Service Discovery
• Service-Oriented Integration
• Security Service Infrastructure
Existing Security Infrastructure
Service-Oriented Security Architecture

Figure 1. Service-oriented security architecture view model

III. SECURITY ARCHITECTURE VIEWS

In this section, a model for service-oriented security architecture specifications is introduced, which is arranged in views. Each view provides details about the security architecture and is tailored for a specific purpose, as it contains apt information for this purpose. We classify three views i) a security engineering view, which provides security information and supporting development artifacts for a software engineering process, ii) a service view, containing information about provided security functionality and usage protocols using the service-oriented paradigm as well as iii) an integration view, which specifies the details of integrating existing security products.

A. *Security Engineering View*

From a security engineering view, a security architecture provides relevant information to a service-oriented development process. The focus thereby lies upon service analysis and discovery as well as service design. Implementation and deployment phases become mostly obsolete, as existing and operated services are used to provide required functionality. Typically only specialized adapter components mapping proprietary interfaces of existing applications to service interfaces need to be developed [27]. In such cases, guidelines can be issued on how security standards are to be used to be compatible with the existing security architecture.

The goal of security analysis is to specify the security requirements of an application. An existing security architecture implements security requirements of previous service-oriented applications. Therefore information about previously identified threats, their corresponding attacks as well as the associated security requirements, implemented to mitigate or prevent the attacks, need to be documented and cataloged. In doing so, developers are supported in analyzing the security requirements of new applications. It would also complement approaches for catalogs of reusable security requirements including possible attacks and threats [24].

During a design phase, security requirements are mapped to appropriate security design. By security design we mean static software structures such as security components and services as well as their dynamic interactions, which are added to the functional application design. A common approach for describing security design is through the use of security patterns, describing best practice solutions to reoccurring security problems. A security architecture should abstract from the implemented security measures, specify them using security patterns and set them into relation to previous security requirements, while keeping the implementation hidden to security service consumers.

Instead of directly attaching a security pattern to the application design, the patterns should be specified separately as part of the security architecture. Semantic annotations can be used to denote the location in the application design, at which security measures are to be adhered. This separates the security design from the application design, according to the separation of concerns paradigm. It also allows the security pattern specification to develop independently from the application design and for better reuse of the security design in different application designs.

Security policies play an important part in a security architecture, as they control the behavior of security components and services [25] and are thus a prime candidate for reuse. An example for security policies are access control policies of which several variations models for specific problems exists [30][31][32]. The security requirements of an application determine the kind of security policies the security architecture has to support. But reusing existing policy models of a security architecture, aides developers by providing a fixed and manageable set of policy models from which an appropriate model can be chosen from.

A security architecture might need to support multiple policy models, depending on different security requirements of different applications. For example, some applications might use a role-based access control policy [30], while others require a more fine-grained access control policy using attribute-based access control [32].

Predefined policies represent the standard security level of an organization, e.g., an enterprise wide constraint on integrity and confidentiality levels of message exchanged between application services. Issuing such policies, relieves developers from choosing an appropriate security policies and makes sure new application comply with the standard security level.

B. *Security Service View*

The security services view is concerned with the provided security services of a security architecture. It acts as a mediator between the abstract security engineering view and the low lever integration view. The abstract security patterns of the security engineering view are mapped to a specific set of security standards and technologies.

While a security architecture aims to centralize most of the security related functionality, there are security-oriented components which are hard to separate from the functional applications. A typical examples are policy enforcement points (PEP), which realize the result of policy decision points (PDP) at a resource [15]. The service view needs to specify how such integration of security-related components and application-oriented services and components is performed. This is usually done using specific interactions protocols, which can be modeled using sequence diagrams of the Unified Modeling Language (UML, [33]).

A service-oriented economy requires interoperability of security services to exchange security information between business partners [34][35]. The usage of XML-based standards such as Security Assertion Markup Language (SAML), eXtensible Access Control Markup Language (XACML), WS-Security [36][37][38] is a good choice for interoperability with external partner but might have a negative effect on performance when used internally. The security services view therefore needs to provide specifications of the available security services, the use of related standards as well as proprietary standards or technologies both for internal use as well as for external use. Additionally the specific use of open standards needs to be documented, as they usually provide a large degree of flexibility.

C. *Security Integration View*

Concerning the integration view, the restructuring of existing proprietary security products to security services, by centralizing security components of existing applications is focused [26]. The other views build upon existing security products, as only their functionality can be provided as services and need to be abstracted from, in order to be used by other views. Due to its technical nature, this view is targeted at an organization's security experts.

We have shown how existing security functionality is integrated into a service-oriented security architecture by implementing web service adapters for a proprietary security product in [27]. Additionally we showed how standardized security policies are mapped to proprietary security product policies automatically. Due to place restrictions, we therefore will not focus on this view and refer to our previous work.

IV. A Service-Oriented Security Architecture based on Web Services

Based on the service-oriented security architecture view model, a concrete security architecture based on web service technology is described in this section. Due to place restrictions, we will concentrate on the access control functionality as well as on the security engineering and the service view, as we discussed the integration of security products in previous work [27]. We assume an existing security product, which provides access control evaluation functionality and which as been adapted to web service technology, if no such interface was provided by the security product itself.

Note also that we do not intend to describe a complete development process for secure service-oriented applications. Instead, we are focusing on single development artifacts that can be provided by examining and describing the security architecture according to the different views as described in the previous section.

A. *Security Engineering Artifacts for Access Control*

The security engineering view remains independent from concrete technologies and abstracts from the web service implementation of the security architecture, similar to the development process of the application services.

Service-oriented application development employs use cases to describe application services and detail their dynamics using process description languages such as the Business Process Modeling Notation (BPMN, [39]) [41]. The security architecture provides information about previously analyzed attacks and threats as well as the formulation of access control security requirements. Currently, misuse cases [19][20] and misuse activities [21][22] are used for this purpose, as they are regarded as initial step in determining security requirements [40].

A misuse case "Unauthorized Access" describes the intentions of an external attacker in accessing a resource, e.g., a service, which he is not allowed to access. The misuse case is stated more precisely through the definition of misuse activities, interacting with the business process of the service (Fig. 2(a)). The access control security requirement associated with the "Unauthorized Access" misuse case is expressed as an UML collaboration diagram (Fig. 2(b)) and can be instantiated using service candidates, which specify the parts of the process to be IT-supported.

The service candidates are the base for the development of the service design, i.e., provided service interfaces implemented by service components. Access control measures for services are provided using an access control service (ACS) and enforcement components, following the policy enforcement point (PEP) and policy decision point (PDP) pattern [15][16], to be reused in service design (Fig. 2(c)). The PEP's task is to query the PDP for access control decisions, which the latter calculates based on existing access control policies, and enforce the decision.

The access control policies used by the security architecture are based on the role-based access control (RBAC) model defined in [30] due to previous requirements (Fig. 2(d)). Developers are offered a common approach for specifying access control policies as well as previously defined roles and their associated entitlements, which can be reused in the development of new applications.

B. *Access Control Service Specification*

Security standards related to web services technology are used to describe the provided access control services. Building upon the service interface specification for the ACS provided in the security engineering view, following artifacts are provided: a web service interface for the access control service (WS-ACS) using the Web Service Description Markup Language (WSDL, [42]), an access control policy model using XACML [37] including policy specification guidelines, a query and response protocol for access control decisions using SAML [36].

As we have chosen to provide a RBAC model for SOA applications, we are using the RBAC profile for XACML [43] as the default specification for XACML access control policies.

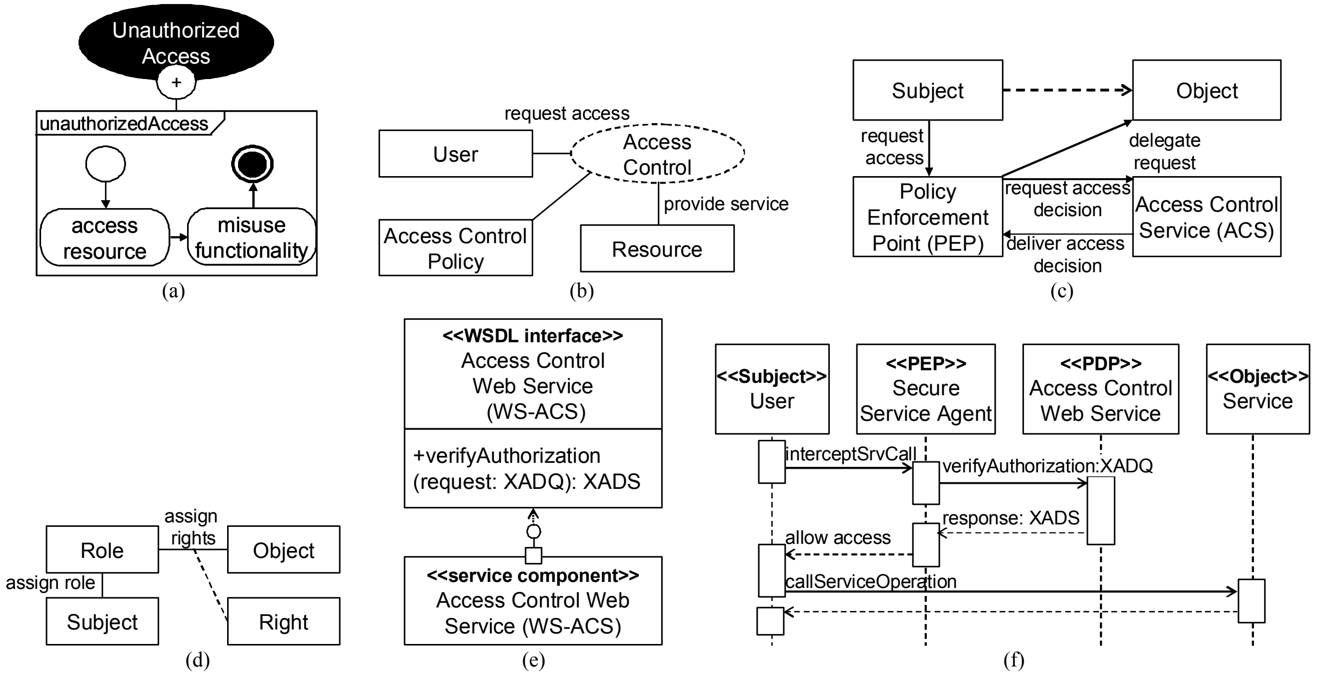

Figure 1. Views on an access control web service specification

While XACML provides access control decision query statements and response statements, it depends on supporting standards to implement a protocol and transport mechanism for them. For this purpose, SAML was chosen as a complement standard [44]. The data flow model of XACML is based on the PEP/PDP pattern discussed earlier. The PEP is implemented using the Secure Service Agent (SSA) pattern [45], which specifies a lightweight component for enforcing access control decisions on web services. The message interception is performed by the SSA by hooking into a web service frameworks' message processing queue. The WS-ACS provides a WSDL interface containing a operation "verifyAuthorization", accepting an XACML Authorization Decision Query (XADQ, [44]) message and returns a SAML statement (XACML Authorization Decision Statement, XADS, [44]) containing the result of the policy evaluation (Fig. 2 (e)).

On service invocation interception, the SSA sends an access control decision request to the WS-ACS using an XADQ request. The request includes the subject, which performs the access, the object, which is to be accessed as well as other contextual information needed to perform policy evaluation. On a positive outcome of the evaluation, the SSA allows the initial service invocation, while on a negative outcome the service access is prevented (Fig. 2 (f)).

V. Case Study: Developing a Secure Campus Navigation System

We used our approach in the development of a secure service-oriented geographical information application for usage at the Karlsruhe Institute of Technology.

A. Case Study Scenario Description

The goal of the application is to support university students and lectures as well as employees and guests in quickly finding and navigating to certain places or people on the campus. A basic functionality of the campus navigation application is to allow users to search for events, persons, rooms and buildings and to display them on a campus map as well as offering a routing service to the destination location.

The services needed for the implementation of the application are provided by different university organization units and are combined in order to create the application. Examples for provided services are a campus and building map service provided by the facility management unit, a staff information service provided by the human resources unit, which provides information about university employees and their corresponding location on the campus side, and event management services provided by different organization units, which provide information about upcoming events such as lectures, practical courses, conferences, etc. on the campus.

B. Approach to Specifying the Security Architecture

As the real-world scenario of the complete university security infrastructure is to complex to be explained in this paper, we will describe the development of the application prototype using the example security architecture described in the previous section. This conforms to the environment used for development and testing the application. We will also stick to the constraint of only discussing the access control aspect of the application.

In order to apply our approach, we first specified a security architecture for the application prototype by choosing existing security products, implementing the core security functionality. As a promising candidate for access control and authentication functionality we chose the open source "Identity Server" (IdS) product from WSO2, as it provided support for most of the relevant web service security standards.

As no previous knowledge for the product was available, the product's documentation was used to provide an abstract specification as required by our view model. The provided functionality, architecture and services were abstracted by matching architecture and security patterns already existent in literature. This process was mostly performed manually and often required deep product knowledge. In the real-world scenario the step was much easier to accomplish, as the required knowledge for the used product was provided by the security experts of the university.

As the IdS supported many security standards required for access control for web services the relevant security standards were used to describe the security service view. Additionally, due to this, it was not necessary to explicitly describe the security integration view.

We then proceeded to examine the existing literature for already described misuse cases, security patterns, etc. to specify the security engineering view of the architecture. We used a variant of the description of authorization misuse cases as provided by [19][20] as well as misuse activities for unauthorized access described in [21][22] to better match with the underlying IdS product. The IdS further implements a variant of the PEP/PDP access control pattern, using a central server as the PDP service component and an authorization module as a PEP component, which is placed in a service bus messaging and communication platform, utilized by the different web services. The choice for role-based access control was also determined by the security product and resembles the actual policy model used in the real-world scenario. The resulting description of the security architecture resembles the specification given in section IV.

C. *Approach to Secure Development*

The development process for the service-oriented application was mainly based on the method presented in [9], but any other approach could have been used as well, as our approach is process independent.

The development started with the requirements analysis of the application by defining use cases representing business services and stating their dependencies. As such a navigation use case was defined, which was related to a map representation, staff information and event management use case. Additionally, the dynamic interactions of the use cases where modeled using BPMN processes.

A security analysis using the previously defined "Unauthorized Access" misuse case and activities (Fig. 2(a)) was performed, which resulted in the informal security policy, that students and unauthorized guests should not be able to search for the location of restricted rooms and areas upon campus, such as the storage facility of chemical substances of the chemistry faculty or the server rooms of the computing centre. A security requirement concerning access control for the search functionality was therefore specified.

The service design phase consisted of several service candidate specifications, which resulted from the use case and process descriptions. The service design was modeled using SoaML [REF] and included a service interface specification as well as a services architecture model, representing the dependencies between services. As several services were already existent, their service interface representations were included in the service design.

As the access control service was predefined by the established security architecture, the access control requirement for the services was modeled by semantically annotating the services using an "authorization" marker tag. This left the service model mostly unchanged, allowing it to focus on its functional representation. The access control policies were described based on the RBAC policy model using a separately modeled UML class diagram. They were also attached to the corresponding services using semantic annotations

After the services were implemented using the provided web service framework of WSO2, the IdS server was configured with an XACML representation of the access control policy. The authorization module in the service bus was than able to determine the entitlements of user requesting access to the services of the application.

VI. Conclusion and Future Work

In this paper, we presented a service-oriented security architecture view model, each of which provides the architectures' intrinsic security information for different purposes by reusing and complementing existing approaches. Three views were identified and described, a security engineering view, providing apt development artifacts, a security service view, describing provided security services, and a security integration view, for the integration and centralization of an organizations security infrastructure. We demonstrated our approach by specifying a security architecture using views and developing a secure service-oriented application using the intrinsic knowledge available by the security architecture.

In the future, we will further research how secure service-oriented applications can be developed using our security architecture view model. So far, we have neglected the issue of introducing new security measures into a security architecture in a methodical way, which is a topic we would like to research next. We also would like to introduce model-driven development techniques into our approach in order to automate the generation of security artifacts, such as security policies.

References

[1] R. J. Anderson, "Security Engineering: A Guide to Building Dependable Distributed Systems", John Wiley & Sons, 2001.

[2] P.T. Devanbu and S. Stubblebine, "Software engineering for security: a roadmap", Proc. Intl. Conf. on the Future of Software Engineering (ICSE '00), ACM Press, 2000, pp. 227-239, doi: 10.1145/336512.336559.

[3] G. McGraw, "Software security: building security in", Addison Wesley, Upper Saddle River, NJ, 2007.

[4] G. McGraw, "Software security", IEEE Security & Privacy, vol. 2 (2), Mar./Apr. 2004, pp. 80-83, doi: 10.1109/MSECP.2004.1281254

[5] A. Toval, J. Nicolás, B. Moros, and F. García, "Requirements reuse for improving information systems security: a practitioner's approach", Requirements Engineering, vol. 6 (4), 2002, pp. 205-219, doi: 10.1007/PL00010360.

[6] C. Gutiérrez, E. Fernández-Medina, and M. Piattini, "PWSSec: process for web services security", Proc. Intl. Conference on Web Services (ICWS '06), IEEE Press, Sept. 2006, pp. 213-222, doi: 10.1109/ICWS.2006.107.

[7] J. Epstein, S. Matsumoto, and G. McGraw, “Software Security and SOA: Danger, Will Robinson!”, IEEE Security & Privacy, vol. 4 (1), Feb. 2006, pp. 80-83, doi: 10.1109/MSP.2006.23.

[8] D. Krafzig, K. Banke, and D. Slama, “Enterprise SOA – Service-Oriented Architecture Best Practices”, The Coad Series, Pearson Education, 2005.

[9] G. Engels et al., “Quasar Enterprise – Anwendungslandschaften serviceorientiert gestalten”, Dpunkt Verlag, 2008.

[10] N. Josuttis, “SOA in der Praxis – System-Design für verteilte Geschäftsprozesse”, Dpunkt Verlag, 2006.

[11] T. Erl, “Service-Oriented Architecture – Concepts, Technology, and Design”, Pearson Education, 2006.

[12] E. B. Fernandez, “A methodology for secure software design”, Proc. Intl. Conference on Software Engineering Research and Practice (SERP’04), 2004, pp. 21–24.

[13] E. B. Fernandez, M. M. Larrondo-Petrie, T. Sorgente, M. Vanhilst, “A Methodology to Develop Secure Systems Using Patterns“, in Paolo Giorgini, „Integrating security and software engineering: advances and future visions”, IGI Press, 2007, pp 107-126.

[14] D. Hatebur, M. Heisel, and H. Schmidt, “Analysis and Component-based Realization of Security Requirements”, Proc. 3rd Intl. Conf. Third Intl. Conf. Availability, Reliability, and Security (ARES ‘08), IEEE Press, Mar. 2008, pp. 195-203, doi: 10.1109/ARES.2008.27.

[15] M. Schumacher, E. B. Fernandez, D. Hybertson, F. Buschmann, P. Sommerlad, “Security Patterns : Integrating Security and Systems Engineering”, Wiley series in software design patterns, Wiley, Chichester, 2006.

[16] Ch. Steel, R. Nagappan, R. Lai, “Core Security Patterns: Best Practices and Strategies for J2EE(TM), Web Services, and Identity Management”, Prentice Hall core series, Prentice Hall PTR, Upper Saddle River, N.J., 2006.

[17] M. Jackson, “Problem Frames: Analyzing and structuring software development problems”, Addison-Wesley, 2001.

[18] N. A. Delessy and E. B. Fernandez, “A pattern-driven security process for SOA applications”, Proc. 3rd Intl. Conference on Availability, Reliability and Security, IEEE press, Mar. 2008. pp. 416-421, doi: 10.1109/ARES.2008.89.

[19] G. Sindre and A. L. Opdahl, “Eliciting security requirements with misuse cases”, Requirements Engineering, vol. 10 (1), Jan. 2005, pp. 34-44, doi: 10.1007/s00766-004-0194-4.

[20] I. Alexander, “Misuse Cases: Use Cases with Hostile Intent”, IEEE Software, vol. 20 (1), Feb. 2003, pp 58-66, doi: 10.1109/MS.2003.1159030.

[21] E. B. Fernandez, M. VanHilst, M. M. Larrondo-Petrie, and S. Huang, “Defining security requirements through misuse actions”, in S. Ochoa and G.-C. Roman, “Advanced Software Engineering: Expanding the Frontiers of Software Technology”, IFIP International Federation for Information Processing series, vol 219, Springer, Boston, 2006, pp. 123-137, doi: 10.1007/978-0-387-34831-5_10.

[22] F. A. Braz, E. B. Fernandez, M. VanHilst, “Eliciting security requirements through misuse activities”, Proc. 19th Intl. Conf. on Database and Expert Systems Application (DEXA ‘08), IEEE press, Sept. 2008, pp. 328-333, doi: 10.1109/DEXA.2008.101.

[23] C. B. Haley, R. Laney, J. D. Moffett, and B. Nuseibeh, “Security requirements engineering: a framework for representation and analysis”, Transactions on Software Engineering, IEEE press, vol. 34 (1), Jan./Feb. 2008, pp. 133-153, doi: 10.1109/TSE.2007.70754.

[24] G. Sindre, D. G. Firesmith, and A. L. Opdahl, “A reuse-based approach to determining security requirements”, Proc. 9th Intl. Works. on Requirements Engineering: Foundation for Software Quality (REFSQ ‘03), 2003, pp 16-17

[25] E. Bertino and L. D. Martino, “A service-oriented approach to security - concepts and issues”, Proc. 11th IEEE Intl. Works. on Future Trends of Distributed Computing Systems (FTDCS '07), IEEE press, Mar. 2007, pp. 31-40, doi: 10.1109/FTDCS.2007.6.

[26] C. Emig, F. Brandt, S. Kreuzer, and S. Abeck, “Identity as a service - towards a service-oriented identity management architecture”, in A. Pras and M. van Sinderen, “Dependable and Adaptable Networks and Services“, Lecture Notes In Computer Science, vol. 4606, pp. 1-8, Springer, 2007.

[27] A. Dikanski, C. Emig, and S. Abeck, “Integration of a security product in service-oriented architecture”, Proc. third Intl. Conf. Emerging Security Information, Systems and Technologies (SECURWARE ‘09), IEEE press, June 2009, pp. 1-7, doi: 10.1109/SECURWARE.2009.8.

[28] M. Hafner and R. Breu, “Security Engineering for Service-Oriented Architectures”, Springer, Berlin, 2009.

[29] D. Linthicum, “Next Generation Application Integration”, Addison-Wesley Information Technology Series, 2004.

[30] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, “Role-based access control models”, Computer, vol. 29 (2), Feb. 1996, pp. 38-47, doi: 10.1109/2.485845.

[31] M. A. Al-Kahtani and R. Sandhu, “A model for attribute-based user-role assignment”, Proc. 18th Annual Computer Security Applications Conference, IEEE Press, 2002, pp. 353-262, doi: 10.1109/CSAC.2002.1176307.

[32] E. Yuan and J. Tong, “Attributed based access control (ABAC) for web services”, Proc. Intl. Conf. on Web Services (ICWS ‘05), IEEE press, July 2005, doi: 10.1109/ICWS.2005.25.

[33] OMG, “Unified Modeling Language 2.3”, 20 Nov. 2010; http://www.omg.org/spec/UML/2.3/, accessed 20 Nov. 2010.

[34] P. J. Windley, “Digital Identity: unmasking identity managemement architecture (IMA)”, O‘Reilly, Beijing, 2005

[35] C. Mezler-Andelberg, “Identity Management - eine Einführung: Grundlagen, Technik, wirtschaftlicher Nutzen”, dpunkt Verlag, Heidlelberg, Germany, 2008.

[36] OASIS, “Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0”; http://www.oasis-open.org/committees/download.php/35711/ sstc-saml-core-errata-2.0-wd-06-diff.pdfx., accessed 20 Nov. 2010.

[37] OASIS, “eXtensible Access Control Markup Language (XACML) Version 2 .0”, http://docs.oasis-open.org/xacml/2.0/ access_control-xacml-2.0-core-spec-os.pdf, accessed 20 Nov. 2010.

[38] OASIS, “Web Services Security: SOAP Message Security 1.1 (WS-Security 2004)”, http://www.oasis-open.org/committees/ download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf, accessed 20 Nov. 2010.

[39] OMG, “Business Process Modeling Notation Version (BPMN) 2.0”, http://www.omg.org/cgi-bin/doc?dtc/10-06-04.pdf, accessed 20 Nov. 2010.

[40] I. A. Tøndel, M. G. Jaatun, and P.H. Meland, “Security requirements for the rest of us: a survey”, IEEE Software, vol. 25 (1), Jan./Feb. 2008, pp. 20-27, doi: 10.1109/MS.2008.19.

[41] M. Gebhart, J. Moßgraber, T. Usländer, and S. Abeck, “SoaML-basierter Entwurf eines dienstorientierten Überwachungssystems”, Gesellschaft für Informatik (GI) Workshop zu SOA und Standardsoftware, Leipzig, Germany, Mar. 2010, pp. 360-367.

[42] W3C, “Web Service Description Language (WSDL) 1.1”, http://www.w3.org/TR/wsdl, accessed 20 Nov. 2010.

[43] OASIS, “Core and hierarchical role based access control (RBAC) profile of XACML 2.0”, http://docs.oasis-open.org/xacml/2.0/ access_control-xacml-2.0-rbac-profile1-spec-os.pdf, accessed 20 Nov. 2010.

[44] Organization for the Advancement of Structured Information Standards (OASIS), “SAML 2.0 profile for XACML”, http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-saml-profile-spec-os.pdf, accessed 20 Nov. 2010.

[45] C. Emig, H. Schandua, and S. Abeck, “SOA-aware authorization control”, Proc. Intl. Conf. on Software Engineering Advances (ICSEA ‘06), IEEE Press, Oct. 2006, pp. 62-62, doi: 10.1109/ICSEA.2006.2613