A Framework for Exploiting Security Expertise in Application Development

Theodoros Balopoulos[1], Lazaros Gymnopoulos[1], Maria Karyda[1], Spyros Kokolakis[1], Stefanos Gritzalis[1], Sokratis Katsikas[1]

[1] Laboratory of Information and Communication Systems Security (Info-Sec-Lab), Department of Information and Communication Systems Engineering, University of the Aegean, Samos, GR-83200, Greece
{tbalopoulos, lazaros.gymnopoulos, mka, sak, sgritz, ska}@aegean.gr
http://www.icsd.aegean.gr/info-sec-lab/

Abstract. This paper presents a framework that enables application developers make use of security expertise. This is succeeded with the help of security ontologies and the employment of security patterns. Through the development of a security ontology developers can locate the major security-related concepts and locate those relevant to the application context. Security patterns provide tested solutions for accommodating security requirements. Finally, the main features of the framework are listed with respect to related work.

1 Introduction

Incorporating security features in the development of applications is an issue that has been attracting the attention of both researchers and developers. To address this issue many solutions have been proposed; some of which are described in section 3 of this paper. However, these solutions are either not always easily applicable in practice, or limited in scope. Therefore, incorporating security requirements in the application development process still remains an open issue. This paper proposes a framework that allows developers to make use of the available security expertise by employing ontologies and security patterns that enable the capture, articulation and reuse of designated solutions to known security issues and requirements.

The paper is structured as follows. Section two reports on the method of work followed, which resulted in the framework proposed in this paper. Section three describes the related work. Section four provides a detailed description of the framework and section five compares the proposed framework to related approaches. Finally, the last section provides our overall conclusions and the directions for future research.

2 Method of work

The framework proposed in this paper is the outcome of a research project exploring ways for the effective introduction of security attributes in the process of application development. Within the research process, security ontologies were first employed in order to explore how they can help developers better understand the application context and communicate with security experts. Results of these efforts have already been published in ([1], [2], [3]). Following this, our research indicated that security patterns would be an appropriate tool for capturing security expertise, and that this can be formalized by employing security ontologies. Thus, based on the ontologies developed, we explored the use of security patterns in the specific application contexts: we designed an appropriate structure for security patterns and a security patterns repository [4]. This paper presents the holistic framework employed, which we believe can provide a useful solution for developers, especially those involved in the development of security critical applications.

3 Related approaches

3.1 UMLsec

UMLsec [5] is a standard UML extension. It allows for the incorporation of security-related information in UML diagrams and supports mechanisms that verify that the security requirements are indeed fulfilled. However, it does not provide step-by-step instructions for reaching this end. The security requirements that can be expressed and validated using UMLsec include confidentiality, integrity, secure information exchange and access control. The major UML diagrams that UMLsec builds upon are the following [6]:

- Class diagrams, which are used to assure that information exchange satisfies the security level stated in the requirements.
- Statechart diagrams, which are used to avoid covert information paths between higher and lower security level entities.
- Interaction diagrams, which are used to verify secure interaction between entities.
- Deployment diagrams, which are used to deal with the security of the physical layer.

3.2 The Ontology-Driven Approach to Information Security

Raskin et al [7] advocate an ontological semantic approach to information security. Both the approach and its resources, the ontology and lexicons, are borrowed from the field of natural language processing and adjusted to the needs of the security domain. This approach pursues the following goals: (i) the inclusion of natural language data sources as an integral part of the overall data sources in information security applications, and (ii) the formal specification of the information security community know-how for the support of routine and time-efficient measures to prevent and counteract computer attacks.

3.3 The Tropos Approach to Modeling Security

Mouratidis et al [8] have presented extensions to the Tropos ontology to enable it to model security issues of agent-based systems. They have introduced the concept of security constraints that allow functional, nonfunctional and security requirements to be defined together, yet being clearly distinguished. They argue that their work makes it easy to identify security requirements at the early requirements stage and propagate them until the implementation stage.

4 Framework for Secure Applications Development

This paper presents a holistic framework, which is depicted in Figure 1, for incorporating security characteristics and accommodating security requirements in application development.

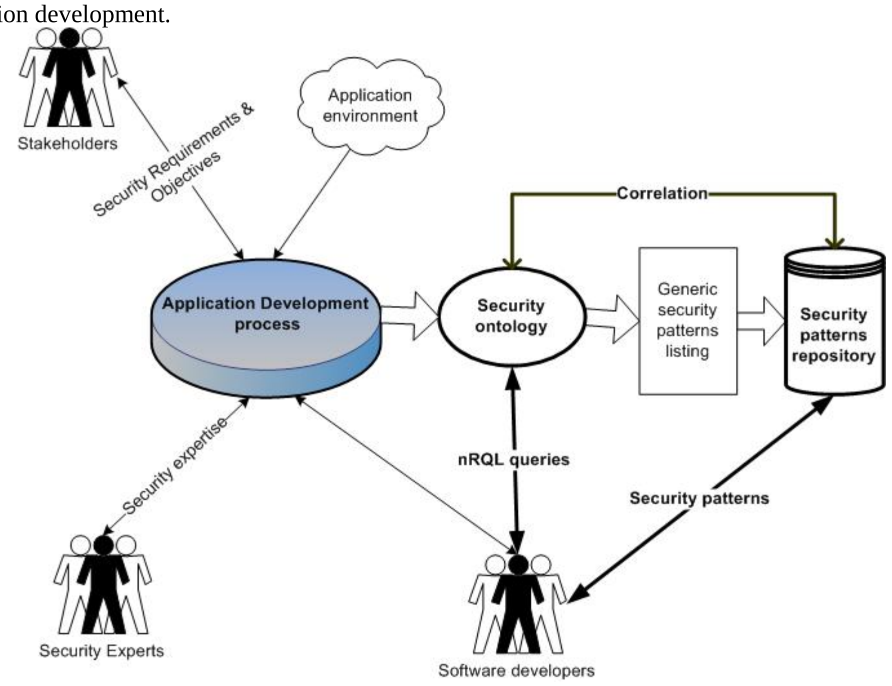

Fig. 1. The framework for secure application development

In [2] we have proposed a methodology for developing security ontologies that can by used to support the process of applications development. In [3] we have presented the use of the developed ontologies in two different application contexts in the area of electronic government. Furthermore, in [4] we have elaborated on the use of security patterns for secure application development and presented the security patterns repository that has been developed throughout this research project.

The framework proposed in this paper constitutes an integrated approach that is addressed to developers that face the need for employing specialized knowledge, and helps them to make use of recorded solutions to known security issues. This holistic framework for application development, builds on the use of ontologies and security patterns, as presented in our previous work that was described above.

Key actors in this framework include (a) the information system stakeholders, i.e. the application users, the administrators and the management, (b) security experts whose knowledge and expertise is needed to enhance the application development process by successfully introducing security features in applications, and (c) the application developers. The latter are the ones that can use this framework for accommodating all different requirements and objectives with regard to security.

Information system stakeholders along with security experts and the software developers set the business and security objectives for the specific application. Existing security expertise is used along with the knowledge of the environment in which the specific application is going to be deployed in order to introduce environment specific security requirements. To achieve this, the basic concepts populating the application context need to be captured and articulated; this is done through the development of the corresponding ontology. Developers can also use existing ontologies or ones that have been developed for similar contexts, as suggested in [3].

4.1 The Security Ontology

An ontology is a description of the entities and their relationships and rules within a certain domain [9]. Ontologies have been widely used within the fields of artificial intelligence, expert systems and the semantic web, mainly for knowledge representation and sharing. Computer programs can use ontologies for a variety of purposes including inductive reasoning, classification, a variety of problem solving techniques, as well as to facilitate communication and sharing of information between different systems. Ontologies are a great tool for defining and communicating the different ways in which people perceive a specific domain. Security ontologies are ontologies covering the domain of security [10].

The Security Ontology depicted in Figure 1 aims at capturing and recording available knowledge regarding business and security objectives of a specific application development environment. The process followed for developing the security ontology, based on the method proposed in [11], was iterative and included four phases: determining competency questions, enumerating important terms, defining classes and the class hierarchy, and finally, the instantiation of the hierarchy.

The competency questions which guided the security development process were loosely structured security oriented questions that the developed security ontology should be able to answer. These questions were taken from typical situations developers face when confronted with security requirements. Next, the most important terms with regard to security were enumerated; the most important of them formed ontology classes; others formed properties of classes and some were not used at all.

In the next phase, the class hierarchy was developed. There are three different approaches in developing a class hierarchy: (a) the top-down approach, where one starts with the definition of the most general concepts of the domain and then goes to the more specialized ones, (b) the bottom-up approach, which starts with the definition of the most specific classes that constitute the leaves of the hierarchy while grouping of these classes into more general concepts follows, and (c) a combination of the two.

To develop the security ontologies presented in [2] and [3] we followed the third of the strategies described previously; our rich set of competency questions fitted well with the top-down approach and resulted in a class hierarchy close the final. Then the bottom-up approach was employed to fit in the remaining concepts.

To examine the rigor of the Security Ontology developed we used queries expressed in the new Racer Query Language (nRQL). This language can be directly used with databases produced by instantiated ontologies through the use of the Protégé software [12] and its Racer interface engine [13]. Further details concerning nRQL queries can be found in [14].

4.2 Security Patterns Repository

Patterns are characterized as solutions to problems that arise within specific contexts [15]. The concept was first used in architecture, but it gained wide acceptance in software engineering with the book "Design Patterns" [16]. The motivation behind the introduction and use of patterns can be summarized as a wish to exploit the possibility of reusability. Thus, patterns are used as "a basis to build on, and draw from, the collective experience of skilled designers" [15].

Security patterns were first introduced by Yoder and Barcalow [16] who based their work on [17]. A security pattern can be defined as a particular recurring security problem that arises in a specific security context, and presents a well-proven generic scheme for its solution [18].

In the proposed framework patterns are used for the same reasons expressed above. Moreover, having in hand the respective ontology – that is a generic description of the security context – developers can easily choose patterns that correspond to that context from a generic list of patterns.

Employing an ontology for the specific application context enables developers to deal with security requirements more effectively. To make use of existing knowledge however, a more concrete solution is needed. Security patterns provide this solution, as they contain both the description of security issues (which can correspond to the requirements) and the indicated method or tool that addresses these issues.

Not all patterns have the same granularity or address security requirements at the same level. For designing the Security Patterns Repository depicted in Figure 1, we have adopted the categorization proposed in [18]. The different categories include:

1. Architectural patterns that refer to the high level software development process.
2. Design patterns that refer to the medium level and refine the components of an application as well as the relationships between them.
3. Idioms are patterns at the lowest level and are related and affected by the programming language that is used each time.

In [4] we have presented a detailed description of the security patterns comprising the Repository built. Table 1 presents a detailed description of each one of them while Table 2 indicates the category they belong to.

Table 1. Security Patterns Comprising the Repository

Pattern Name	Description of the pattern
Authentication	This pattern allows users to access multiple components of an application without having to re-authenticate continuously. It incorporates user authentication into the basic operation of an application.
Password authentication	This pattern concerns protection against weak passwords and automated password guessing attacks.
Credentials propagation	This pattern requires that users' authentication credentials are verified by the database before access is provided.
Cryptographic storage	This pattern uses encryption for storing sensitive or security-critical data in the application.
Encrypted Communications	This pattern uses encryption for the secure transmission of sensitive or security-critical data over a network.
Session Management (protection of specific session)	This pattern provides that users cannot skip around within a series of session regarding a specific function (task) of an application. The system will not expose multiple functions but instead will maintain the current task that the users desire.
Hidden implementation	This pattern limits an attacker's ability to discern the internal workings of an application—information that might later be used to compromise the application.
Partitioned application	This pattern splits a large, complex application into two or more simpler components. Thus, dangerous privileges are restricted to a single, small component. Each component has tractable security concerns that are more easily verified than in a monolithic application
Patching	During the application lifetime, bugs and vulnerabilities are discovered; patches must be provided to address these issues.
Logging - auditing	Applications and components offer a variety of capabilities to log events that are of interest to administrators and other users. If used properly, these logs can help ensure user accountability and provide warning of possible security violations

Table 2. Patterns' Categorization

Pattern Name	Pattern category
Authentication	Architectural
Password authentication	Architectural
Credentials propagation	Architectural
Cryptographic storage	Design
Encrypted Communications	Design
Session Management	Idiom
Hidden implementation	Architectural
Partitioned application	Architectural
Patching	Design
Logging - auditing	Design
SandBoxing	Idiom

5 Comparison with other approaches

The proposed framework has the following features:

- It captures the knowledge of security experts, and aims to use it to address the needs of the software developer. Other approaches, such as [5] and [7] are not focused on the software developer, but on the security expert.
- It employs an ontology to model information. This ontology deals with objects of higher structure than other approaches (such as [7] or [8]), namely security patterns, thus being able to suggest solutions and promote reusability more effectively.
- It proposes a different instantiation of the ontology per security context. This allows it to model the fine details that a general ontology such as the one proposed in [7] is much more difficult to capture.
- It is not limited in context, unlike approaches such as [4], which is dedicated to agent-based systems.
- It can be utilized to search among the possible solutions for the one that best fits the context, unlike approaches such as [5] that are utilized to validate an already chosen solution.

6 Conclusions and Further Research

This paper presents a combined approach to incorporating security knowledge and expertise in the application development process. It advocates the development and employment of security ontologies, which (a) can facilitate the communication among the different parties involved, i.e. developers, security experts and the application stakeholders and (b) provide a way to capture and describe the basic security-related concepts, e.g. the security requirements the application should comply with. Moreover, the use of ontologies helps aggregate different views on the security features of the application. However, the use of ontologies has some limitations, since their construction is hard and time-consuming, and there is no standardized procedure to follow. Finally, the use of security patterns, through the creation of a repository, enables developers use standard solutions for accommodating these requirements.

Up to now, this research project has produced a set of security ontologies for similar application environments that mostly relate to electronic government, as well as a series of patterns, covering different aspects of security requirements in applications, that correspond to the ontologies. The next steps in the research process include covering different domains (e.g. the domain of health applications), as well as designing the basic mechanisms for adding functionality to the security patterns repository, and more specifically enabling their management (adding, comparing, deleting, association etc.).

Acknowledgments

This work was co-funded by 75% from the European Union and 25% from the Greek Government, under the framework of the "EPEAEK: Education and Initial Vocational Training Program—Pythagoras".

References

1. Balopoulos T., Dritsas S., Gymnopoulos L., Karyda M., Kokolakis S. and Gritzalis S., "Incorporating Security Requirements into the Software Development Process', in Proceedings of the 4th European Conference On Information Warfare And Security, University of Glamorgan (ECIW 05), UK, 11-12 July 2005.
2. Dritsas S., Gymnopoulos L., Karyda M., Balopoulos T., Kokolakis S., Lambrinoudakis C., Gritzalis S., "Employing Ontologies for the Development of Security Critical Applications: The Secure e-Poll Paradigm", in Proceedings of the IFIP I3E International Conference on eBusiness, eCommerce, and eGovernement, October 2005, Poznan, Poland, Springer Verlag.
3. Karyda M., Balopoulos T., Dritsas S., Gymnopoulos L., Kokolakis S., Lambrinoudakis C., Gritzalis S., "Using Security Ontologies for the development of secure e-Government applications", in Proceedings of the DeSeGov'06 Workshop on Dependability and Security in eGovernment (in conjunction with ARES 2006 1st International Conference on Availability, Reliability, and Security) A. Tjoa, E. Schweighofer (Eds.), April 2006, Vienna, Austria, IEEE Computer Society Press
4. Gymnopoulos L., Karyda M., Balopoulos T., Dritsas S., Kokolakis S., Lambrinoudakis C. and Gritzalis S. "Developing a Security Patterns Repository for Secure Applications Design" in the Proceedings of the 5th European Conference on Information Warfare and Security (ECIW 2006), National Defence College, Helsinki, Finland.
5. Jurjens, J. (2001) Towards development of secure systems using UMLsec, Lecture Notes in Computer Science, 2029:187.
6. Stevens, P. et al (2000) Using UML. Addison-Wesley.
7. Raskin, V., Hempelmann, C., Triezenberg, K., and Nirenburg, S.: Ontology in Information Security: A Useful Theoretical Foundation and Methodological Tool. In Viktor Raskin and

Christian F. Hempelmann, editors, Proceedings of the New Security Paradigms Workshop, New York. ACM, (2001).

8. Mouratidis, H., Giorgini, P., Manson, G.: An Ontology for Modelling Security: The Tropos Project. Proceedings of the KES 2003 Invited Session Ontology and Multi-Agent Systems Design (OMASD'03), United Kingdom, University of Oxford, (2003).
9. Gruber T. R., “Toward principles for the design of ontologies used for knowledge sharing,” Presented at the Padua workshop on Formal Ontology, March 1993.
10. Filman R. and Linden T., Communicating Security Agents, In Proceedings of The Fifth Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, Stanford, CA, USA, 1996, pp. 86-91.
11. Noy, N.F. and Mc Guinness, D.L. "Ontology Development 101: A Guide to Creating Your First Ontology", Stanford Knowledge Systems Laboratory Technical Report KSL-01-05. 2001.
12. Protégé, http://protege.stanford.edu/
13. Racer Inference Engine, http://www.sts.tu-harburg.de/~r.f.moeller/racer/
14. The New Racer Query Language, http://www.cs.concordia.ca/~haarslev/racer/racer-queries.pdf
15. Schumacher M., Fernandez-Buglioni E., Hybertson D., Buschmann F., and Sommerland P., Security Patterns: Integrating Security and Systems Engineering, John Wiley & Sons, 2006.
16. Yoder, J., Barcalow, J.: Architectural Patterns for Enabling Application Security. In: Proc. 4th Conference on Pattern Languages of Programs (PLoP 1997), Monticello, IL, USA, 1997.
17. E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns – Elements of Reusable Object-Oriented Software, Addison-Wesley Professional, 1995.
18. Schumacher M. (2003), Security Engineering with Patterns : Origins, Theoretical Models, and New Applications, Paperback, 2003