Available online at www.sciencedirect.com

SciVerse ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

A methodology for integrating access control policies within database development

Jenny Abramov[a,b], Omer Anson[b], Michal Dahan[a], Peretz Shoval[a], Arnon Sturm[a,*]

[a] Department of Information Systems Engineering, Ben-Gurion University, Markus Campus, Beer-Sheva 84105, Israel
[b] Deutsche Telekom Laboratories (T-Labs), Ben-Gurion University, Markus Campus, Beer-Sheva 84105, Israel

ARTICLE INFO

Article history:
Received 30 January 2011
Received in revised form
8 January 2012
Accepted 17 January 2012

Keywords:
Authorization
Access control
Database design
Security patterns
Domain analysis
FOOM
ADOM
UML
Security

ABSTRACT

Security in general and database protection from unauthorized access in particular, are crucial for organizations. While functional requirements are defined in the early stages of the development process, non-functional requirements such as security tend to be neglected or dealt with only at the end of the development process. Various efforts have been made to address this problem; however, none of them provide a complete framework to guide, enforce and verify the correct design of security policies, and eventually generate code from that design.

We present a novel methodology that assists developers, in particular database designers, to design secure databases that comply with the organizational security policies that are related to access control. The methodology is applied in two main levels: organizational level and application development level. At the organizational level, which takes place before the development of a specific application, organizational policies are defined in the form of security patterns. These patterns encapsulate accumulated knowledge and best practices on security related problems. At the application development level, the data-related security requirements are defined as part of the data model. The security patterns, which have been defined at the organizational level, guide the definition and implementation of the security requirements. The correct implementation of the security patterns is verified during the design stage of the development process, before the automatic generation of the database code. The methodology is supported by a CASE tool that assists its implementation in the various stages.

1. Introduction

Data is a most valuable asset for an organization, as its survival depends on the correct management, security, and confidentiality of the data (Dhillon, 2001). In order to protect the data, organizations must secure data processing, transmission and storage. In spite of that, even nowadays, organizational systems are being developed with minor treatment of security aspects; system developers tend to neglect dealing with security requirements, or deal with them only at the end of the development process.

It has been recognized that security must be treated from the early stages of the software development lifecycle, and various efforts have been made to address this need. Examples for studies that deal with incorporation of security aspects within the development process vary from UML extensions such as UMLsec (Jürjens, 2005) and SecureUML (Lodderstedt et al., 2002; Basin et al., 2006), to methods for designing

* Corresponding author.
E-mail addresses: jennyab@bgu.ac.il (J. Abramov), oaanson@gmail.com (O. Anson), dahanmic@bgu.ac.il (M. Dahan), shoval@bgu.ac.il (P. Shoval), sturm@bgu.ac.il (A. Sturm).

0167-4048/$ – see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cose.2012.01.004

secure databases such as Fernández-Medina and Piattini (2005). Yet, such studies mainly provide guidelines about the way security should be handled within certain stages of the software development process, or address specific aspects of security. To the best of our knowledge, no existing methodology provides a complete framework that both guides and enforces organizational security policies on a system design, and then generates executable code from that design.

We propose a comprehensive methodology that provides such a framework. The proposed methodology deals with the organizational level and with the application development level. At the organizational level, the organizational security policies are defined by security officers and domain experts, in the form of security patterns. At the application development level, the system analysts/designers create a conceptual data model and a functional model of the application, which include security requirements. These models are further refined and adjusted following the security patterns, and then are verified compared to the organizational level security policies. Finally, the code of a secure database schema is generated. The whole process, at both levels, is supported by a software tool that has been developed as part of this study.

Although many different security aspects should be dealt with when developing systems, in this study we deal mainly with the database aspect, while in follow-up work we also deal with other aspects (e.g., system behavior/functionality, and users' interfaces). In this study we focus on secure database development for the following reasons: 1) Data are the most important resource, that have to be protected from undesired access and manipulations; 2) Usually, when designing the security within the application layer, the connection to the database is done using a single user, thus eliminate the option of auditing and identification of malicious activities within the database; and 3) Databases can be accessed directly and through various applications, thus it is important to secure the database itself. Nevertheless, there are other important security aspects that should be addressed; the presented methodology can be extended to deal with such aspects as well.

The contribution of this work is the following: We provide a mechanism to precisely specify security patterns using well-known and widely-used modeling techniques, and add transformation rules to the security patterns. The precise definition of the patterns enables the enforcement of their content over application designs, whereas the transformation rules allow the automatic generation of the desired artifacts based on the specified patterns. Furthermore, we deal with conformance checking of application designs vis-à-vis the organizational security patterns.

The rest of this paper is structured as follows: Section 2 reviews related studies; Section 3 presents the background related to the proposed methodology; Section 4 describes the methodology; and Section 5 concludes and set plans for future research.

2. Related studies

Over the years, many methods and techniques have been proposed to incorporate security aspects within the development process of information systems. The studies in this survey are organized in two categories: a) general security specification techniques; and b) access control specification techniques. Finally, we discuss the limitations of the existing methods.

2.1. *General security specification techniques*

UML use cases (UC) are a common method to model functional requirements; thus, it is natural to expect that this method will be extended to deal with security requirements. One such extension is "misuse cases" (Sindre and Opdahl, 2005). Misuse cases describe how different kinds of actors might attack or misuse the system and their relationships with the desired functionality. Gomaa and Shin (2009) suggested another way to define security requirements using UCs. They proposed to separate the functional UCs from the security UCs, as opposed to the above use and misuse cases. The security UCs handle only the security requirements, and are later on combined with the functional UCs.

Several specification techniques for presenting security policies in a model-driven software development process have been proposed. UMLsec (Jürjens, 2005) was one of the first to integrate security with UML. UMLsec extends UML to enable specifying security concerns in the functional model. It uses standard UML extension mechanisms, stereotypes with tagged values to formulate the security requirements, and constraints to check whether the security requirements hold in the presence of particular types of attacks. In the context of access control, UMLsec provides a notation to represent RBAC (Role-Based Access Control) policies and to specify guarded access.

Secure Tropos (Mouratidis and Giorgini, 2007) is a security-oriented extension to the Goal-Driven Requirements Engineering methodology, which enables to model security concerns of agent-based systems by allowing functional and non-functional requirements to be defined together, yet being clearly distinguished. Secure Tropos methodology is applied in four main phases: (1) During the early requirements analysis phase, functional as well as security requirements are analyzed in terms of constraints imposed by the stakeholders of the system. Then, secure goals and entities are identified, which guarantee the satisfaction of the constraints. (2) During the late requirements analysis phase, further analysis of the system within its operational environment, together with relevant functions and security specification, is performed. (3) During the architectural design phase, the architectural style of the system is defined. In this stage, the designer describes in detail all of the actors with respect to their goals and tasks, identifies and assigns agent capabilities while considering the identified security requirements, and transforms the requirements to a design with the aid of security patterns. (4) During the detailed design phase, the designer specifies each architectural component in further detail. A major limitation of Secure Tropos is that it mainly focused on the phases of early and late requirements analysis, but it does not provide adequate support to the design of security policies.

To overcome this limitation, Mouratidis and Jürjens (2010) combined Secure Tropos and UMLsec to create a structured methodology for secure software development that supports all software development phases. Since Secure Tropos' main focus is on requirements analysis, and UMLsec's main focus is on security analysis and design, the two approaches