

# Uniform Application-level Access Control Enforcement of Organizationwide Policies

Tine Verhanneman Frank Piessens Bart De Win Wouter Joosen  
Katholieke Universiteit Leuven, Department of Computer Science  
Celestijnenlaan 200A, 3001 Leuven  
{tine,frank,bartd,wouter}@cs.kuleuven.be

## Abstract

*Fine-grained and expressive access control policies on application resources need to be enforced in application-level code. Uniformly enforcing a single policy (referred to as the organizationwide policy) in diverse applications is challenging with current technologies. This is due to a poor delimitation of the responsibilities of application deployer and security officer, which hampers a centralized management of a policy and therefore compromises the uniformity of its enforcement.*

*To address this problem, the concept of an access interface is introduced as a contract between an organizationwide authorization engine and the various applications that need its services. The access interface provides support for the central management of the policy by the security officer. By means of a view connector, the application deployer ensures that each application complies with this contract, so that the policy can be enforced.*

## 1. Introduction

Many applications require the enforcement of an expressive access control policy, which, for example, takes into consideration application state. In the context of an organization, where one single organizationwide policy needs to be enforced, support should be provided to administer the policy centrally and to enforce it uniformly in the various applications deployed within the organization.

Various stakeholders are involved in the enforcement of an access policy. The *security officer* manages the policy centrally. The *application deployer* tunes the access control enforcement for each application so that the policy can be enforced. The access control decision itself can be delegated to an organizationwide authorization engine, which may be developed independently of a particular application setting and provided by an *authorization engine provider*.

The *application developer* provides the application logic.

In this paper, the observation is made that the delimitation of the responsibilities between the security officer and application deployer, is poorly supported by current technologies: In reality, the application deployer bears complete responsibility for the uniform enforcement of the policy. This renders it hard to manage the policy centrally, especially if this policy is liable to frequent changes.

The contribution of the paper consists in providing an abstraction layer, named *access interface*, which captures the requirements an application needs to fulfill so that the organizationwide policy can be enforced. This access interface, for example, includes explicitly the additional information that is needed to evaluate an access request. The access interface abstracts from application-specific details by including only information that is relevant for access control. It can therefore be specified by the security officer, who is responsible for the definition and centralized management of the policy.

The application deployer binds the access interface to each application by means of application-specific *view connectors*. A view connector specifies (1) how the application fulfills the requirements that are put forward in the access interface, and (2) how access requests within the application are translated to the access interface. A prototype has been implemented as an extension of an aspect-oriented application container, whereby the view connector acts as a deployment descriptor.

The remainder of the paper is organized as follows. Section 2 motivates the need for an intermediary abstraction layer, illustrated by a case in the health care application domain. In Section 3 the access interface approach is presented, followed by a discussion in Section 4. The prototype is discussed in Section 5. Section 6 gives an overview of related work and conclusions are drawn in Section 7.

















