An Architecture for Secure Policy Enforcement in E-Government Services Deployment

Nikolaos Oikonomidis, Sergiu Tcaciuc, and Christoph Ruland

Institute for Data Communication Systems, University of Siegen
{nikolaos.oikonomidis, sergiu.tcaciuc,
christoph.ruland}@uni-siegen.de

1 Introduction

Citizens interact at regular intervals with municipalities or municipal organizations. Public administrations offer a variety of services like requests/processing of certificates and (local) tax payment. An effective and efficient service provision brings benefits to both municipalities and the involved citizens/customers of a particular service. Due to the fact that exchanged data in forms and documents may contain private or/and sensitive data, it is imperative to introduce security mechanisms that guarantee to citizens a trustworthy means of communication via a network that may be insecure, such as the Internet. Further, cross-border services involve different municipalities and other public authorities in the processes. The described work is derived from research for "eMayor", a project funded by the EU committee. eMayor addresses the specific audience of Small and Medium sized Governmental Organizations (SMGOs) across Europe. The project looks especially at transactions that are performed on a European level. This paper focuses on an architecture for secure policy enforcement within eGovernment platforms, such as the eMayor platform.

2 Secure Policy Enforcement

The approach chosen for modeling the overall architecture of eMayor relies on the Reference Model of Open Distributed Processing (RM-ODP) [1]. At first, the identified requirements together with the legal frameworks formed the Enterprise Viewpoint. This viewpoint resulted into a specification of a community of the platform users and the respective business objects that derive from the community specification. The identified scenarios have been converted into processes. The Information Viewpoint has presented and analyzed various information objects that exist in the eMayor context and the relations between them. Additionally, since Information Objects pass from various states through their life-cycle, their respective state transitions have been specified as well. The interaction between system components on the functional level and their respective interfaces have been described in the Computational Viewpoint. Engineering and Technology Viewpoints have been placed in the implementation phase.

The system design resulted into the specification of an architecture as a set of modules, each one comprising certain functionalities. *User Interface* handles the interaction of the user with the eMayor platform, required for the actual processing of the

J. Dittmann, S. Katzenbeisser, and A. Uhl (Eds.): CMS 2005, LNCS 3677, pp. 284 – 285, 2005.

service. *Service Handling* represents the core of the system and has dependencies to all other modules. *Format Transformation* is responsible for transforming legal documents from a country-bound local format to a universal format for use within the eMayor environment and vice versa. *Content Routing* provides the routing functionality for forwarding requests and legal documents from one municipality to another. *Municipal Systems Adaptation* is the linking point to the existing (legacy) systems of the municipalities. *Persistent Storage* handles data storage to databases. *Output Notification and Printing* provide support for notification and printing services. Finally, *Policy Enforcement* encapsulates the enforcement of a series of functionalities which are defined in municipal policies. Such functionalities include, e.g., auditing, access control, digital signature verification and other security mechanisms. Within the Information Viewpoint, a policy information object has been specified. Such a policy object represents the constraints, conditions, and rules that have to be respected and enforced by the platform. *Security Services Policy* implements the security services that are required. *Access Control Policy* regulates access control to the requested municipal services. *Audit Policy* controls how actions are recorded in the system for auditing purposes. *Policy Subject* represents the entity which will invoke one or more *Policy Action* objects. A *Policy Action* may take effect on a *Policy Object*. The result of a *Policy Action* or even one ore more *Policy Objects* are related to the *Policy Target*. One or more *Policy Pre Condition*, *Policy Condition* and *Policy Post Condition* objects control the invocation of one or more *Policy Action* objects depending on the policy type. A subject of research is modeling of policies that control the secure execution of the municipal services. In other terms, a *Service Execution Policy* will define the steps that have to be taken during the system's operation regarding the required security services and the pre-/post conditions that should be fulfilled.

Policy Enforcement is implemented in the policy enforcement module, which resides in the eMayor platform of each municipality. Components of the policy enforcement module comprise different sets of functionalities in order to enforce the appropriate policies. *Policy Enforcement Management* is the component that exposes the module's functionality as a set of "enforcer" interfaces. *Policy Evaluation* component contains all elements which are responsible for taking a decision if a request or functionality complies to the appropriate policy, whereas *Policy Retrieval* component queries the respective policy repositories and retrieves the appropriate policy. The model of the components and their communication within the policy enforcement package derives from the XACML specification [2]. The current research objective is to provide an extension to XACML for enforcement of other types of policies apart from access control policies.

References

1. Information technology, Open Distributed Processing - Reference Model: Architecture, ISO, 1996
2. Core Specification, eXtensible Access Control Markup Language (XACML) Version 2.0, OASIS, 2005