Chapter 14
Enforcing ASTD Access-Control Policies with WS-BPEL Processes in SOA Environments

Michel Embe Jiague  
Université de Sherbrooke, Canada, & Université Paris-Est Créteil Val-de-Marne, France

Marc Frappier  
Université de Sherbrooke, Canada

Frédéric Gervais  
Université Paris-Est Créteil Val-de-Marne, France

Régine Laleau  
Université Paris-Est Créteil Val-de-Marne, France

Richard St-Denis  
Université de Sherbrooke, Canada

ABSTRACT

Controlling access to the Web services of public agencies and private corporations depends primarily on specifying and deploying functional security rules to satisfy strict regulations imposed by governments, particularly in the financial and health sectors. This paper focuses on one aspect of the SELKIS and EB^3SEC projects related to the security of Web-based information systems, namely, the automatic transformation of security rules into WS-BPEL (or BPEL, for short) processes. The former are instantiated from security-rule patterns written in a graphical notation, called ASTD that is close to statecharts. The latter are executed by a BPEL engine integrated into a policy decision point, which is a component of a policy enforcement manager similar to that proposed in the XACML standard.

INTRODUCTION

In some business sectors, information systems (IS) are governed by internal organization policies and government laws. Access control is widely used in IS to enforce such policies as well as to prevent breaches of data confidentiality and integrity security. More precisely, user access to data and functionalities are filtered, based on well-defined policies. Role-based access control (RBAC), a methodology that associates user identities with the data and/or functionalities through their role,
is the most commonly implemented solution. It does not, however, solve new problems found in today’s common service-oriented architecture (SOA) applications. These applications are not solely “one user centric” and implement workflows that may involve interactions with different users. With respect to workflows, RBAC has little expressiveness power. As an example, RBAC frameworks cannot implement separation of duty (SoD) properties.

A substantial part of the EB³SEC (EB³SEC stands for EB³Secure) and SELKIS (SELKIS is an acronym for SEcure heaLth care networKs Information Systems) projects consists in developing a prototype of a policy enforcement manager (PEM) for distributed IS executed in an SOA environment as Web services (WS). Our approach focuses on three levels of access control: the data level, the RBAC level, and the process level. In this paper, we propose an automatic implementation of a significant part of the enforcement framework derived from an access-control policy expressed in a high-level language. This high-level language is formal, powerful enough to implement common properties encountered in security policies, and can also express many sorts of constraints. Its implementation relies on a translation algorithm, which produces an executable BPEL process from a formal specification of an access-control policy. Overall, the enforcement framework follows architectural guidelines proposed in the XACML standard.

The rest of this paper is organized as follows. The next section introduces Algebraic State Transition Diagram (ASTD), the formal notation used for specifying security rules and presents four security-rule patterns: permission, prohibition, obligation, and SoD. The ASTD notation provides for combining state-transition diagrams with process algebra operators. This high-level notation is appropriate for specifying security rules at the process level and is independent of any implementation environment. The following section presents the architecture of SOA applications targeted by our projects and the three levels of granularity we identified previously. Enforcement framework components are described and typical message-exchange scenarios between the principal components are depicted. The sequel details a translation schema that transforms an ASTD specification into a BPEL process along with its WSDL interface and XSD type definitions, which are deployed in an SOA environment so that they constitute the core of the policy decision point (PDP). This translation schema is mechanizable as far as the security rules obey the aforementioned patterns. A safe translation procedure then replaces an error-prone development phase. The last sections describe strongly related work and point out differences with some aspects developed in this paper and conclude with ongoing and future aspects of this work.

**EXPRESSING SECURITY RULES WITH AN ASTD**

In most security frameworks, a security policy is a combination of many security rules. Researchers and security practitioners (Basin, Burri, & Karjoth, 2009; Konopacki, Frappier, & Laleau, 2010a, 2010b; Yao, Moody, & Bacon, 2001) have considered the following categories for security rules:

- **Permission** which authorizes actions to be executed;
- **Prohibition** which forbids actions to be executed;
- **Separation of duty** which expresses the fact that a set of tasks cannot be executed by the same users or roles;
- **Obligation** which forces a user to perform an action sometime in the future after he has performed a specific action. In other words, two distinct actions must be performed by the same user.