An Event-Based, Role-Based Authorization Model for Healthcare Workflow Systems

Vassiliki Koufi, Flora Malamateniou, Eleni Mytilinaiou, and George Vassilacopoulos

Department of Digital Systems, University of Piraeus,
80, Karaoli & Dimitriou Str, 18534, Piraeus, Greece
{vassok,flora,emytilin,gvass}@unipi.gr

Abstract. Authorization and access control is of primary importance to workflow healthcare environments. Although task dependencies in the workflow give rise to the need for a specific ordering of task executions, it is authorization that determines who can execute the various tasks that comprise the workflow and what information can be accessed during task executions. Furthermore, a challenge of workflow security is to enforce the least privilege principle (i.e. users must be allowed to receive the least possible permissions required to perform a task) throughout workflow execution in order to reduce the risk of compromising information integrity during task executions. However, adherence to the least privilege principle often requires the enforcement of dynamic, contextual constraints so that authorizations for access to data during task executions are granted to and revoked from users dynamically. This paper discusses an event-based, role-based workflow authorization model and mechanism that addresses this issue. In particular, the model augments the capabilities of traditional role-based access control (RBAC) models by allowing user roles to change dynamically during workflow execution based on the occurrence of specific events, in order to prevent users from acquiring unnecessary privileges during workflow task executions and, hence, synchronizing authorization flow with the progression of the workflow.

Keywords: healthcare, workflow systems, role-based authorization, least privilege principle, ECA rules.

1 Introduction

Role-based access control (RBAC) has been strongly featured in workflow security research as a foundation on which several access control models have been built, each covering its own range of applications in its own way [11, 12]. Moreover, RBAC has been widely adopted by the industry given that most of the Business Process Management (BPM) Systems are already based on this paradigm [13, 14]. The basic premise underlying the implementation of an RBAC model in a workflow environment is that roles are viewed as sets of permissions on task executions and data object accesses during task execution and that an access control policy is essentially defined as a pair of mappings: a mapping of users to roles and a mapping of roles to permissions. Hence, each role is assigned certain permissions on task executions and data object

M. Szomszor and P. Kostkova (Eds.): E-Health 2010, LNICST 69, pp. 221–228, 2011.

accesses at build time and the permissions granted to users at run time emanate from their roles and may be subject to both static and dynamic constraints which are usually imposed to express authorization policies [2].

One important requirement that is essential for healthcare workflow systems to provide a high level of system security is the least privilege principle [1, 5]. Using RBAC to enforce the least privilege principle would essentially involve allowing users to assume the absolute minimum roles (or permissions assigned to the role) required for effective task executions and synchronizing the validity of relevant authorizations with the time intervals specified for task executions [2, 4, 5]. However, RBAC models are not capable of encompassing certain application-level contextual constraints which further limit the role-based access capabilities and privileges of users, specified at workflow build time, based on the overall context associated with any workflow instance. Hence, conventional access control may be too coarse for certain healthcare workflows and what is needed is a fine-grained access control that is based on the user's working context. Even the same user may have different data access needs based on the task instance he/she is working on.

This paper describes a dynamic, contextual workflow authorization model with regard to application data accesses required during workflow task executions. The model retains the advantages of broad, role-based permission assignment and administration as in RBAC, while it provides the flexibility for granting (revoking) fine-grained, context-dependent roles to (from) users dynamically at workflow run time to ensure a tight matching of roles to actual need. The granting, tracking and revoking of these roles is automated and synchronized with the progression of the workflow to ensure that users are awarded only the privileges needed for completing a job effectively and are prevented from acquiring privileges for longer time intervals than needed [2, 4].

2 Motivating Scenario

The basic motivation for this research stems from our involvement in a recent eHealth project concerned with the development of a web-based ePrescribing service. The stringent security needs of the system motivated this work and provided some of the background supportive information for developing the authorization system prototype.

Suppose a healthcare delivery situation concerned with drug prescriptions. A physician uses an ePrescribing service which is interfaced to a Personal Health Record (PHR) system, accesses the summary record of his/her current patient, and selects one or more drugs based on information regarding eligibility status of the medication list covered. Upon selection of a drug by the physician, the ePrescribing application performs clinical validation checks (e.g. regarding drug interactions, patient allergies and medication history) to either clear the prescription or return alert information to physician. Then, the physician sends the e-prescription to a designated data center where the whole ePrescription activity is captured. Finally, the patient or a delegated person thereof collects the prescribed drugs from a pharmacy of his/her choice.

Figure 1 shows a high-level view of the ePrescribing process. In this process two healthcare professional roles are involved: the physician and the pharmacist. Table 1 shows an extract of workflow authorization requirements regarding task execution and related data access privileges assigned to these roles, respectively.

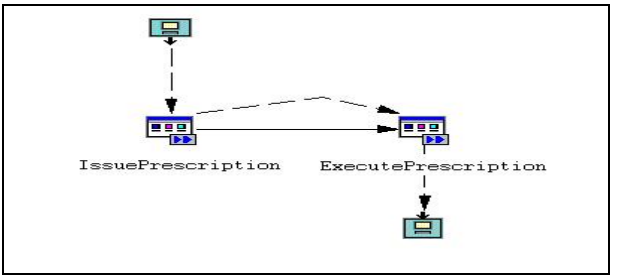

Fig. 1. A high level model of an ePrescribing process

Table 1. Extract of authorization requirements for the healthcare process of Figure 1

1.	Physicians may issue medication prescriptions on their patients. (*IssuePrescription*)
1.1	Physicians may write medication prescriptions for their current patients.
1.2	Physicians may edit medication prescriptions for their current patients before sent.
1.3	Physicians may send medication prescriptions for their current patients.
1.4	Physicians may cancel medication prescriptions for their current patients after sent.
1.5	Physicians may read patient records of their patients only.
2.	Pharmacists may execute medication prescriptions for patients. (*ExecutePrescription*)
2.1	Pharmacists may read patient medication prescriptions issued by physicians.
2.2	Pharmacists may read the past drug profile of patients they are about to execute medication prescriptions.
2.3	Pharmacists may execute medication prescriptions for patients.
2.4	Pharmacists may update patient drug profile with medication prescriptions executed.

From a role-based, workflow authorization perspective, the healthcare process of Figure 1 surfaces several requirements with regard to task execution and associated data accesses. These requirements include the following:

- **Task execution -** Task execution specified by a role-to-task permission assignment may be further restricted dynamically during ePrescribing process execution and be a subset of the authorized role holders. For example, prescription may only be issued by physicians who are contracted to the patient's insurance organization (having access to the "IssuePrescription" task) and may only be executed by qualified pharmacies (e.g. primary care prescriptions may only be executed by a pharmacy within a health district and hospital prescriptions may only be executed by the hospital pharmacy).
- **Data access -** Some role holders are allowed to exercise a set of permissions on certain data objects only and/or for a limited duration. For example, during the execution of the "IssuePrescription" task, a physician may be allowed to read patient records and issue prescriptions only for his/her patients and the permission to read patient records may be revoked upon successful task execution. Moreover, pharmacists may have the authorization to modify prescriptions with regard to brand name but not with regard to drastic substance of each medicine.

The above requirements suggest that certain data access permissions depend on the ePrescribing process execution context. In particular, contextual information available at access time, like location or user/patient relationship, can influence the authorization decision that allows a user to perform a task and access associated data objects. This enables a more flexible and precise authorization policy specification that incorporates the advantages of having broad, role-based permissions across process tasks and data object types, like RBAC, yet enhanced with the ability to simultaneously support the above requirements. In addition, the model should not incur any significant administrative overhead and should be self-administering to a great extent.

3 Contextual Authorization

Typically, role-based authorizations with regard to task execution and data access are specified during workflow build time when the exact user-to-role and role-to-permission assignment relationships are decided. These authorizations are static in nature since, once defined, they are valid for all workflow instances until they are manually revoked by the security administrator [2]. However, static authorizations constitute a basic set of authorizations and are not flexible enough to capture security policies of a healthcare organization on workflow execution that incorporate the context in which task instances are assigned to users and performed by users as workflow progresses. Dynamic, context-aware authorizations are intended to provide the required flexibility by taking this context into account when deciding on the permission(s) that should be granted to healthcare process participants at run time. Hence, these authorizations are bound to specific task instances (e.g. incorporate constraints on the data content, the user identity, the valid time and the location of attempted accesses).

From a workflow security perspective, context can be defined as any information that is available at run time and is considered relevant to assigning (executing) a task instance to (by) a user and to granting (revoking) associated data access permissions to (from) the user [6]. Thus, every workflow is associated with a context which is defined as a set of context types that may correspond to domain-dependent and domain-independent concepts. *Domain-dependent context types* are those related to the particular workflow under study (e.g. corresponding to concepts for user and patient). On the other hand, there are also *domain-independent context types* (e.g. corresponding to concepts for time and location).

By analyzing the workflow authorization requirements, workflow application designers determine which context types are relevant to each task. For example, with regard to the healthcare process of Figure 1, the workflow authorization requirements related to the "IssuePrescription" task specify that "A physician can access the records of his/her patients only" and "A physician can issue a drug prescription for his/her patients only". From these requirements there is a need for a concept that describes the "attending" relationship between a physician and a patient to enable a physician to access the record and to issue a drug prescription for his/her patient.

Roles may be assumed as named collections of capabilities and privileges which represent organizational agents intended to perform business functions [9]. These roles may be termed "*functional roles*" and define the division of work and the lines of authority based on job functions and seniority. In a typical workflow environment,

functional roles encapsulate sets of permissions on task executions and data object types, they are assigned to users at workflow build time, through a many-to-many relationship, and they may be activated at run time, possibly with regard to domain-independent contextual constraints such as spatio-temporal.

To encapsulate sets of permissions that take into account the domain-dependent context associated with each task, the concept of "*contextual role*" is defined. Thus, contextual roles encapsulate sets of permissions which are tailored to run time access needs of users so that users are provided with tight, just-in-time permissions (i.e., contextual roles encapsulate the minimum sets of permissions required for effective task executions). The contextual role concept is also used as a mechanism that associates users with contexts in a similar manner as the role concept is used as an intermediary between users and permissions.

A contextual rule relates information from contexts in logical expressions that specify an access control policy to protected objects. They also enable the definition of parameterized expressions whose arguments are evaluated when a task assignment decision is requested or a data access during task execution is attempted.

To alleviate users from the burden to change roles at workflow run time, as required by the current work context, and to reduce the administrative overhead imposed, a rule-based approach should be adopted that enables automatic role changes (e.g. from functional to contextual) on the occurrences of specific events (e.g., specific points in time, activations of workflow or task instances). Role-based access control approaches for workflow systems based on active rules have also been reported in several studies in the past [5, 8, 10]. Thus, two kinds of events may be assumed, namely *task activation events* and *task termination events*, which are referred to as *role change events*. A task activation event occurs when a user pulls a task instance from the work list to start working on it and has the effect of granting a contextual role to the user so that the user receives only the data access permissions required for an effective task instance execution. Thus, the user holds both his/her functional role(s) and the contextual role just granted. A task termination event occurs upon successful execution of a task instance and has the effect of revoking the user's contextual role granted earlier so that the user's data access permissions encompassed in the contextual role are revoked. Hence, the user is left with his/her functional role(s) only. An occurrence of a role change event would automatically trigger the processing of a relevant rule defined in the event-condition-action (ECA) format [8].

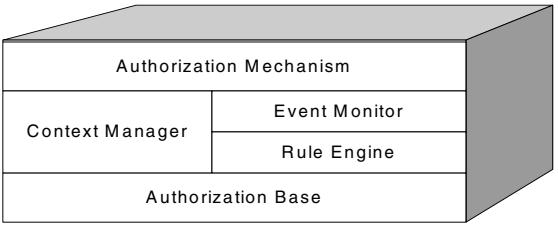

Fig. 2. Authorization architecture

4 Authorization Architecture

Based on the authorization concepts described above, an experimental workflow authorization system has been developed. As shown in Figure 2, the architecture of the prototype authorization system designed and implemented consists of the following components:

- **Authorization Base -** It records users, roles (functional and contextual) and permissions as well as user-to-role and role-to-permission assignments. Role-based authorizations are stored in the form of authorization rules with regard to task executions and data object accesses. In addition, the authorization base contains role change event occurrences as well as the contextual constraints defined.
- **Event Monitor -** It records event occurrences into the authorization base and passes the event occurrence data to the rule engine that triggers the appropriate contextual role granting/revocation rule. Since events are associated with tasks, the event monitor is informed on the occurrence of an event by associating tasks with contextual role granting/revocation operations. Thus, a contextual role granting/revocation operation is executed upon successful execution of the associated task operation. The event monitor is informed on event occurrences when a user attempts to initiate or terminate a task execution.
- **Rule Engine -** It is responsible for (a) storing contextual role granting and revocation rules in ECA format, (b) determining the appropriate rule when information on the occurrence of an event is received from the event monitor and (c) triggers the contextual role granting (revocation) action when the condition specified in the rule is satisfied (i.e. records/deletes an entry in the user-to-contextual role relationship table of the authorization base).
- **Context Manager -** On the occurrence of an event, it determines the current work context and communicates the collected information to the rule engine which uses it when triggering the relevant ECA rule to change a functional to contextual role and vice versa.
- **Authorization Mechanism -** It accesses the permission relationships of the authorization base to enforce access control on users holding a role at the time of an attempted task execution or data access. Thus, on an attempted task execution or data access the authorization mechanism mediates between role holders and tasks (data objects) to determine whether the requested action should be permitted or denied.

The above architecture implements a centralized policy service for workflow elements enabling healthcare providers to delegate authorization decisions to an authorization policy server deployed in the workflow environment.

5 Prototype Development

To illustrate the features of the above authorization model, a Business Process Execution Language (BPEL) engine was used to incorporate the authorization system designed as a separate module. This engine was used for an experimental implementation of a workflow-based service that drew on the business rules and authorization requirements of the healthcare process depicted in Figure 1.

Referring to the healthcare process of Figure 1, two web services have been developed which are concerned with issuing prescriptions ("Issue_Prescription") and with executing prescriptions ("Execute_Prescription"). In addition, another web service developed ("PHR_service") is concerned with accessing the patient's personal healthcare record by either the physician or the pharmacist. Thus, basically, the tasks of the ePrescribing process depicted in Figure 1 have been implemented as web services, including the healthcare record access sub-task.

Assume there is a physician wishing to issue a prescription for one of his/her patients. Then, the following actions are performed with regard to security: On attempting to execute the task "IssuePrescription", a task activation event occurs which is identified by the event monitor and is recorded into the authorization base while event occurrence data is passed to the context manager and to the rule engine. Then, the context manager evaluates the contextual parameters specified to determine the current context that constraints authorization rights (e.g. the patient currently attended by the physician) and passes this information to the rule engine. The rule engine identifies and triggers the relevant ECA rule to grant the contextual role "attending physician" to the user, by inserting an appropriate entry into the authorization base, and passes the contextual role to the authorization mechanism. The authorization mechanism consults the authorization base and grants to the user the restrained permissions for executing the task, invoking the relevant web service (i.e. "Issue_Prescription") and invoking "PHR_service" to enable access to the relevant patient's data if required. Similar activities take place when a pharmacist attempts to execute the task "ExecutePrescription" in order to execute a prescription for a patient.

6 Concluding Remarks

Workflow systems can offer great benefits in the development of modern process-oriented services spanning through healthcare organizational boundaries. However, for workflow systems to reach their full potential in automating healthcare processes, authorization and access control mechanisms must be in place that can conveniently and cost effectively regulate user access to information while providing confidence that security policies are faithfully and consistently enforced. In particular, if adherence to the least privilege principle is considered a prominent feature of the service, the authorization mechanism needs to provide tight, just-in-time permissions so that the appropriate users get access to specific objects only when they provide their services. To this end, it is required that permissions are neither granted "too early" nor revoked "too late" and that a tight matching of permissions to actual need is ensured. The dynamic workflow authorization model presented in this paper provides an approach towards achieving this goal, in the context of RBAC models, by introducing the contextual role concept and allowing authorizations encapsulated in these roles to be granted to and revoked from users at workflow run-time based on the occurrence of specific events. Potential weaknesses of both the proposed model and system may be revealed during system evaluation. This is one of the tasks to be undertaken in the near future and may result to alterations in the proposed approach.

References

1. Abrahams, A.S., Eyers, D.M., Bacon, J.M.: An Event-Based Paradigm for E-Commerce Application Specification and Execution. LNCS Information Security, Technical Report 6(2), 59–68 (2001)
2. Atluri, V.: Security for Workflow Systems. LNCS Information Security, Technical Report 6(2), 59–68 (2001)
3. Botha, A.R., Eloff, H.P.: A Framework for Access Control in Workflow Systems. Inform. Manage. Comput. Se. 9(3), 126–133 (2001)
4. Casati, F., Castano, S., Fugini, M.: Managing Workflow Authorization Constraints through Active Database Technology. Inf. Syst. Front. 3(3), 319–338 (2001)
5. Dey, A.K., Abowed, G.D.: The Context Toolkit: Aiding the Development of Context-Aware Applications. In: Human Factors in Computing Systems (CHI 1999), Pittsburgh, PA, May 15-20, pp. 434–441 (1999)
6. Goh, A., Koh, Y.K., Domazet, D.S.: ECA Rule-based Support for Workflows. Artif. Intell Eng. 15, 37–46 (2001)
7. Lenz, R., Reichert, M.: IT Support for Business processes. In: van der Aalst, W.M.P., Benatallah, B., Casati, F., Curbera, F. (eds.) BPM 2005. LNCS, vol. 3649, pp. 354–363. Springer, Heidelberg (2005)
8. Malamateniou, F., Vassilacopoulos, G., Tsanakas, P.: A Workflow Based Approach to Virtual Patient Record Security. IEEE T. Inf. Technol. B. 2(3), 139–145 (1998)
9. Zhou, X., Wang, Z.: An Access Control Model of Workflow System Integrating RBAC and TBAC. Integration and Innovation Orient to E-Society 2, 246–251 (2008)
10. Xing, G., Xue, S., Liu, F.: Design of Role-Based Security Access Control Model in the Workflow. In: 1st IEEE International Conference on Information Science and Engineering (ICISE 2009), Nanjing, Jiangsu China, December 26-28, pp. 1711–1715 (2009)
11. Zhao, H., Fang, Z., Xu, P., Zhao, L., Liu, J., Wang, T.: An Improved Role-Based Workflow Access Control Model. In: 5th International Conference on Information Technology: New Generations (ITNG 2008), Las Vegas, Nevada, USA, April 7-8 (2008)
12. DeCarlo, A.L.: Dynamic Business Process Management (BPM): Applying a Role-Based Approach to Business Process Management, InformationWeek, Business Technology Network (2009), `http://business-agility.techweb.com/articles/02042009.jhtml`
13. Byron, D.: Role-based Business Process Management: A Good Way to Think of BPM, IT Business Edge (2009), `http://www.itbusinessedge.com/cm/blogs/byron/role-based-business-process-management-a-good-way-to-think-of-bpm/?cs=34553`