Available online at www.sciencedirect.com
SCIENCE DIRECT®
Information Systems 32 (2007) 365–384
www.elsevier.com/locate/infosys

DW-RBAC: A formal security model of delegation and revocation in workflow systems

Jacques Wainer[a,*], Akhil Kumar[b], Paulo Barthelmess[c]

[a]*Institute of Computing, State University of Campinas, Campinas, 13083-970 SP, Brazil*
[b]*Smeal College of Business, Penn State University, University Park, PA 16802, USA*
[c]*Department of Computer Science and Engineering, Oregon Health & Science University, Beaverton, OR 97006, USA*

Received 31 August 2004; received in revised form 12 July 2005; accepted 2 November 2005

Recommended by M. Weske

Abstract

One reason workflow systems have been criticized as being inflexible is that they lack support for delegation. This paper shows how delegation can be introduced in a workflow system by extending the role-based access control (RBAC) model. The current RBAC model is a security mechanism to implement access control in organizations by allowing users to be assigned to roles and privileges to be associated with the roles. Thus, users can perform tasks based on the privileges possessed by their own role or roles they inherit by virtue of their organizational position. However, there is no easy way to handle delegations within this model. This paper tries to treat the issues surrounding delegation in workflow systems in a comprehensive way. We show how delegations can be incorporated into the RBAC model in a simple and straightforward manner. The new extended model is called RBAC with delegation in a workflow context (DW-RBAC). It allows for delegations to be specified from a user to another user, and later revoked when the delegation is no longer required. The implications of such specifications and their subsequent revocations are examined. Several formal definitions for assertion, acceptance, execution and revocation are provided, and proofs are given for the important properties of our delegation framework.

Keywords: Workflow; Access control; Delegation; Role-based access control; Security

1. Introduction

Despite various advances in workflow technology, current workflow systems still do not handle delegation well. In its simplest form, a user who has been assigned a task and is unavailable to perform it for any reason (such as leave of absence, sickness, etc.) should be able to delegate it to another user. If such support is not provided, the task will not get done.

The role-based access control model (RBAC) (for example [1]) is receiving attention as a systematic way of implementing the security policy of an organization. It groups individual users into roles that relate to their position within an organization and assigns permission to various roles according to their stature in the organization. Roles are generic

*Corresponding author. Tel.: +55 19 37885871.
E-mail addresses: wainer@ic.unicamp.br (J. Wainer), akhilkumar@psu.edu (A. Kumar), paulo@cse.ogi.edu (P. Barthelmess).

0306-4379/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.is.2005.11.008

terms like manager, vice-president, etc. and anybody in a role can perform certain tasks assigned to him or her.

The term *delegation* is usually employed in the security literature to describe transfer or inheritance of rights from some user to a machine, that then acts as a surrogate for that user (as in an ATM transaction, for instance). Only recently researchers are starting to recognize the importance of introducing delegation into the RBAC framework. The two significant research efforts that we are aware of in this direction are those of Barka and Sandhu [2,3] and of Yao, Moody and Bacon [4]. The work of Barka and Sandhu allows a role to delegate to another role, and also considers multi-step delegations and revocations. Yao et al. [4] introduces the notion of an appointment whereby a user can appoint another user to perform a task.

RBAC features are increasingly being supported in commercial database systems such as Informix, Sybase and Oracle [5], and the term *grant* is used to refer to the assignment of privileges to users and role. Moreover, *grant* is itself a right that can be conferred. In this way, delegation can be implemented in a database system. However, such support is still limited and does not permit very fine-grained control in a dynamic environment.

In RBAC, in addition to roles, users and privileges, there is also a notion of sessions. Thus, a user may log into different sessions with different roles that she is entitled to play. For example, in one session Mary may be logged in as a cashier and in another as an accounts manager. In workflow applications, the concept of a session is less clear. Instead workflow systems have a notion of *cases*, corresponding to the processing of a specific instance, such as Beth's expense reimbursement claim for travel to Chicago, or Carl's auto accident claim. In this context, privileges must be case-specific, i.e., a user may have permission to perform a task for a certain case but not be allowed to perform the same task for another case. For example, Beth must not be the *requester* and *approver* for *the same reimbursement*, but, of course, Beth may be the approver for Carol's request, and may herself be the requester of a different reimbursement process. Thus, Beth may be *at the same time* requesting her reimbursement and approving Carol's, but that is acceptable if these roles are being played in different *reimbursement cases even within the same session*. Previous work by the authors [6] extends RBAC to accommodate case-based privileges in the context of workflow systems. The present paper is an attempt to include delegation into such a framework. This work extends the ideas presented in [7], which discuss some of the ideas of a fine-grained delegation/revocation framework in RBAC. In this work those ideas have been extended into a workflow domain.

This paper is organized as follows. Section 2 describes briefly the motivation behind the W-RBAC model, an extension to the RBAC model to deal with workflow systems. Section 3 describes the key intuitions behind our model of delegation. Then Section 4 discusses the formal aspect of the assertion of a delegation. Section 5 discusses the intuitive and formal aspects of revocation of delegations. Section 6 proposes extensions to our framework to allow richer kinds of delegations, and Section 7 describes a proof of concept implementation. Then, Section 8 discusses related work, while Section 9 gives the conclusions of this work.

2. W-RBAC and workflow/permission system

2.1. Workflow management systems and RBAC

Workflow management systems allow for the definition and enactment of business processes. A workflow system stores definitions (or schemas) of processes in terms of their tasks definitions, users that should perform tasks, usually given in terms of roles, and a partial ordering of tasks that establishes constraints on task execution sequences. Additional constraints may also be imposed on the ordering. After a workflow process W is defined, instances of this process (also known as *cases* in the workflow literature) may be created and are managed (or *enacted* in the workflow literature) by the workflow system. At any time, zero or more instances of a workflow process might be being enacted.

The focus of the present paper is on the set of users that can perform workflow tasks. Besides timely instantiation of tasks (control flow aspect), one of the main duties of a workflow system is to determine who among the users are the most appropriate to execute each task, as well as determining an order of preference if more than one user fits the requirements for execution (resource allocation aspect). In most commercial workflow systems, once the set of potential executors is determined, either the system announces to all of them that there is work to be done, and one of the users will accept the work, or some more