Active Authorization as High-level Control

Daniel Cvrček
PhD. student on DCSE, Brno University of Technology
Božetěchova 2, 61266 Brno
Czech Republic
cvrcek@dcse.fee.vutbr.cz

Key words: authorization system, workflow, active authorization

Abstract: The paper introduces several issues that have one common target - secure cooperation of autonomous information systems. We show that Active authorization model may be an abstract layer that allows simple, efficient and secure management of heterogeneous system's security properties.

1. INTRODUCTION

Nowadays commercial sector demands possilibity to specify well-defined tasks that represent usual business processes. It means that users may work with intuitive tasks like *process customer order* or *prepare contract of insurance* and so on.

The tasks' definitions may be divided into two abstract levels. The lower level contains tasks representing atomic (relatively simple) actions. The higher level comprises workflows, tasks that may be invoked by external subjects (e.g. users) or other applications.

Commercial environment implies existence of users that can not be fully trusted and will never be experts in security. Communication among physically distant information systems is usually performed via insecure channels and quality of access controls and used models differs. Those facts demand existence of very tight security permissions that allow application of *need-to-know* and *need-to-do* principles. Ideal seems to be authorization

model that preservers and joins one security state with each particular task. General properties of such a model have been given in [6].

2. OVERVIEW OF SECURITY IN DISTRIBUTED SYSTEMS

The first thing we have to do for solving security issues is to split the distributed system into *homogenous* (from the security point of view) parts that are centrally administered. We shall call them *autonomous information system* or *s-node*.At this moment, we may solve secure cooperation among s-nodes. The identified problems are:

a) Access control in s-node - this problem comprises access control to resources local in s-node and is solved by access control model implemented in the local platform. Each autonomous system may have implemented other access control model.
b) Global administration of system - distributed system has to solve problems with heterogeneity of its s-nodes and enforce uniform administration of security properties.
c) Flow control - we are talking about systems that allow space distributivity of computational tasks. Those tasks use data with different sensitivity, stored on many s-nodes. Flow control enforces uniformity throughout the system - *reference monitor* [1].

We may control several different types of resource accesses (with increasing abstraction).

1. Access to resources on s-node
2. Access to resources in workflow
3. Access to workflows
4. Data flow in workflows

It is clear that *discretionaryon control* is sufficient for the lowest level of access control. The mandatory or some other type of axiomatic access control (e.g. RBAC [2,3,4]) has to be on the other side used for access control to workflows and their resources (e.g. RBAC) when a common security policy is to be enforced.

3. COOPERATION WITH ACCESS CONTROL

Cooperation between global authorization model (AAM) [5] see items 2-4 above and local authorization models (item 1) is the crucial aspect of successful enforcement of access control rules throughout the distributed system. We do not know about any work that solves problems rising from

this cooperation. Existing papers explicitly use only RBAC model for determination of subjects able to activate tasks.

We try to generalize cooperation between AAM and s-nodes' access control systems. The basic condition is made by introducing general function $\Phi(S', P_I') \rightarrow S_0'$ that allows determination of subjects - initiators, able to run a task, from the set of all subjects according to the set of privileges necessary for initialization of the task.

We do not say which way is the set obtained. We do not say what model is used for that purpose. It may be a function that finds all users that are specified in a given UNIX group or a function cooperating with RBAC model that uses much more sophisticated ways for the purpose.

3.1 Examples of Local Access Control Models

There are vital differences among different access models. Because of the lack of space we only name the most important. DAC models such as HRU, Take-Grant, Acten. MAC models Bell-LaPadula, Biba or Dion and also object oriented models ORION, Iris.

Generally, one may say that basic DAC (discretionary access control) models do not offer any properties that can be used for generalization of access control and there are problems with centralized administration. To solve those problems we have to create layer isolating system with DAC model from external subjects (users).

MAC (mandatory access control) models contain general axioms that may be used for common security. Those axioms express certain general rules that may be used for centralized administration.

The first problem that has to be solved is general architecture of authorization system as a whole. We are interested in heterogeneous distributed systems composed from s-nodes that are able to communicate among themselves. Each s-node contains resources it is able to work with (files, peripheral devices, ...). It also has a system that manages access to resources (operational system, database management system) and there is an access (or authorization) control model that is used for managing access to resources. The control model has to be a part of authorization system.

3.2 Basic Layering of Authorization System

It is useful to create a basic layering of authorization system based on the architecture of the distributed system.

We have already said that s-nodes contain (or use) access control model. This is the first layer and we call it *Local Access Control*. This layer should

be able to create new users or user groups and provide instruments that allow authentication of more abstract systems (or its users).

Definition 1: Local Access Control $\boldsymbol{L_i}$ is n-tuple $\boldsymbol{L_i}=\{S_i, O_i, A_i, f_i\}$ that consists of set of users S_i, set of resources (or objects) O_i set of access modes A_i and authorization function f_i that is defined as follows:

$f_i: S_i \times O_i \times A_i \rightarrow \{True, False\}$

The function f_i is able to decide access requests. ■

The second layer of our architecture should be able to convert global definition of privileges into a form applicable in Local Access Controls and vice versa. This layer shall be called *Conversion Layer*. This is the first element that creates some general (global) framework.

Definition 2: Conversion Layer represents two functions σ_i' and σ_i'' that take set of users and set of resources from the underlying Local Access Control $\boldsymbol{L_i}=(S_i, O_i, A_i, f_i)$, set of resource categories C^k and returns subset of users that are authorized to access specified resources.

$\sigma_i'' : C^k \times S_i \rightarrow S_i$

uses σ_i' to translate resource categories C^k into the set of resources O_i^k

$\sigma_i' : C^k \rightarrow O_i^k$

and of course the function f_i to determine the final subset of users with privileges to access all elements from the set of resources O_i^k. ■

Next layer should allow specification of tasks executable in the distributed system. We distinguish two types of tasks.

- Tasks that are executed on one s-node and by one user. We call those tasks *atomic tasks*. The description of those tasks is dependent on the particular s-node.
- Tasks that may be executed on several s-nodes and/or by several users. We call them *workflows* and their specification is *platform independent*.

Atomic tasks are described by means of given s-node. Particular layer is *Atomic Task Manager*. The last one, *Workflow Manager* is completely s-node independent. It allows its implementation by means satisfying execution on various systems. Workflow Manager may be split into two parts, one concerning *static authorizations* of workflows (authorizations that are specified during workflow creation) and one concerning *dynamic authorizations*.

3.3 Communication of Security Layers

Authorization system of any distributed system has to be *active*. We do not know all users of the distributed system in one central place and s-nodes have to be able to determine whether and where are users able to run particular task or workflow step.

How to perform a workflow? Imagine that someone on s-node S_i has started a workflow and executed the first step of it. We need to find all s-nodes that are potentially able to execute next step of the workflow.

(1) We have to address all s-nodes in the distributed system. (2) Each s-node identifies users that are able to continue the workflow and wait until one of the identified users initiates execution of the task. (3) When the *active* user is on the s-node S_j, then S_j responds to S_i. (4) S-node S_i recalls its request on all s-nodes except S_j. And (5) execution of the workflow moves to the new s-node S_j and the *active* user may execute next step.

The most important is step (2). Communication among layers of the authorization system on particular s-nodes is performed here. The Workflow Manager receives the first impulse. It has to ask the Atomic Task Manager, if the particular task is defined there. In the case of success, the Atomic Task Manager has to determine set of users authorized (static rights) to run the task (either directly with Local Access Control or in cooperation with Conversion Layer). The same task but with the dynamic authorization is done by Workflow Manager, Conversion Layer and Local Access Control. There are received two sets of users. Their intersection is a set of users authorized to execute the workflow's task.

3.4 The Conversion Layer

Conversion Layer is the place, where two considerably different models are in touch. Very important is to find criterion for general specification of authorization requirements. The following possibilities were identified.

- Hierarchy in the organization.
- Name of subject (user).
- Privilege for data access (analogy with MAC).
- Reference to a common hierarchy (absolute or relative (from-to)).
- Reference to another predefined role (group) structure.
- Types of resources that have to be accessed.

The classification we shall use has to be very stable and must be applicable for all s-nodes. We have assumed that the most general and stable classification should be based on resource categories (defined according to data content) that form a non-hierarchical set of elements C^k that depends on the environment.

When using just category of resource, subject is able to perform all possible operations over accesible resources. We have got no information to restrict set of operations. The restriction in this direction is enforced through the tasks' definitions.

Resource categorization is the fixed point that allows global definition of workflows. All task definitions use categorization (or classification) to specify security requirements for data access (resources and workflows). During execution of particular task step are categories converted into form that the s-node's Local Access System is able to use to determine authorized users and to determine needed resources (functions Φ and Θ). Atomic Task Manager performs this conversion especially in Conversion Layer and partially.

4. CONCLUSION

We have proposed base ideas of the problem of unification of security administration and secure cooperation among autonomous information systems. The offered approach consists of two cornerstones. The first one is existence of uniform security classification (or categorization) of resources. This fact constitutes fixed point that consolidates security administration of s-nodes. The second one is design of authorization system in such a way that allows separation of particular platforms from active subjects (especially users). This structure allows uniform definition of workflows on any s-node in the distributed system.

The result is the architecture of authorization system that allows secure execution of workflows, centralized administration of the whole distributed system and decentralized definition of workflows.

5. BIBLIOGRAPHY

[1] D.E. Bell and L.J. LaPadula, Secure computer systems: Unified exposition and multics interpretation Technical Report MTR-2997, Bedford (MA), The Mitre Corporation, March 1976.

[2] E. Bertino, E. Ferrari, and V. Atluri, A Flexible Model Supporting the Specification and Enforcement of Role-based Authorizations in Workflow Management Systems, Proceedings of the Second ACM Workshop on Role-Based Access Control (Fairfax, VA), November 1997.

[3] R. S. Sandhu et al., Role-based Access Control Models, IEEE Computer, February 1996, p. 38-47.

[4] R.S. Sandhu, Role Hierarchies and Constraints for Lattice-Based Access Controls, Proc. Fourth European Symposium on Research in Computer Security, September 1996.

[5] D. Cvrcek, Mandatory access control in workflow systems, Knowledge-based Software Engineering - Proc. of the JCKBSE - Conference, 2000, pp. 247-254.

[6] R.K. Thomas and R.S. Sandhu, Towards a Task-based Paradigm for Flexible and Adaptable Access Control in Distributed Applications, Proceedings of the Second New Security Paradigms Workshop (Little Compton, Rhode Island), IEEE Press, 1993.