MULTIPLE AUTHORIZATION

A Model and Architecture for Increased, Practical Security

Gerald Vogt
Munich Network Management Team — University of Technology Munich
Oettingenstr. 67, 80538 Munich, Germany
Gerald.Vogt@informatik.uni-muenchen.de

Abstract: Security of systems and management infrastructure is crucial for a successful, reliable and safe use. Most currently deployed systems are based on simple subject/object-relations where control of access happens. This has the drawback that any access decision occurs just on the behalf of the single subject accessing. In this paper we describe an extended access control model that allows to include authorization of multiple subjects thus overcoming this drawback while still focusing on practical aspects of simple integration in many of the existing systems.

Keywords: Security Management, Access Control, Authorization

1. INTRODUCTION

Access control is the part of a security system that actively maintains security checking authorizations and thereby controlling access to protected resources and functions. The main parts involved in access control are shown in figure 1. The active entity accessing the protected operation is called the *subject* for this access. The subject is usually linked with a real person who has been identified during authentication. The *object* of an access can be any kind of resource or function that is accessible and protected for security reasons. A subject performs an *operation* on an object by calling a method/sending a message/making a procedure call, depending on the underlying paradigm and passing all necessary parameters that define the details of the operation.

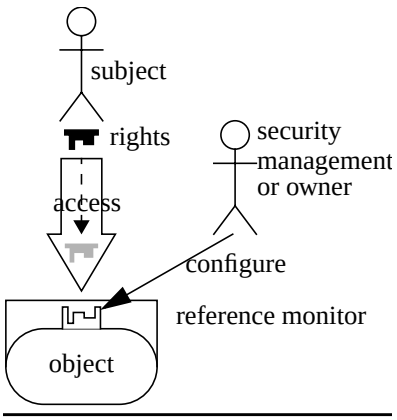

Figure 1. Simple access control

The *reference monitor* is the part of the security system that actively enforces access control. It checks the authorization of the accessing subject and decides whether the subject is allowed to execute the operation or not. The identity of the subject and/or other concepts like tickets or capabilities can be used to determine the rights and to prove the required authorization. It is the duty of security management to assign the necessary rights to subjects depending on the tasks they have to work on. In addition, many systems also allow an owner of a resource to assign rights, e.g., the owner of a file grants access to other subjects. The responsibility of the owner often coincides with security management.

The access control model described in this section very much follows the idea of discretionary access control (DAC) [1], one of the traditional access control models, that is the foundation of many current operating systems like UNIX and languages like Java. The discussion sections briefly covers some other access control models and how they relate to multiple authorization presented in the following section.

2. MULTIPLE AUTHORIZATION

One of the limiting factors of current access control systems is the dependence on a single subject for authorization. It is usually not possible to directly involve other entities, i.e., other people in a particular access control decision. This, however, would be reasonable if the access involves an object that is critical and severe effects could occur unless done properly. In the simplest case of two subjects this is the ‘four-eyes’ principles from real-world scenarios when something must be done in mutual agreement of two persons. *Multiple Authorization* as presented in this section is a practical mean to integrate this idea in existing systems. It is an extension of the simple, single-subject authorization model underlying most currently used systems. Instead of depending on a single subject, it allows to include several subjects where necessary (Figure 2). In the following, we use fractional rights to demonstrate the concept, e.g. 1/3 for a third of the full right. Yet, it is not limited to fractions and arbitrary constraints could be used here as well, though they certainly add some complexity.

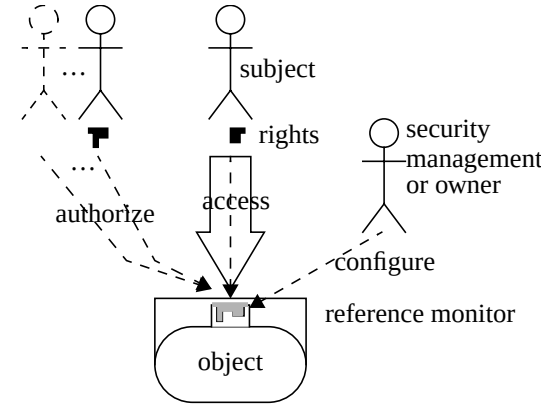

Figure 2. Access control with multiple authorization

First of all, when an *accessing subject* accesses an operation it expresses its will to execute it. If the subject does not have sufficient rights (e.g., only 1/3) to execute the operation alone, the authorization is in *incomplete* state. Other subjects in the new role of the *authorizing subject* can now supplement (e.g. with another 1/3) until authorization is *complete* (i.e. the sum is >= 1). Supplementing authorization of an

operation is a new system function that does not exist in simple authorization. As the authorizing subject authorizes an operation access of a different subject, it needs information about the operation in question before it can decide what to do. This requires the ability to *inspect* the incompletely authorized operation with all its parameters as well as the progress of authorization until now and further context or state information available. The authorizing subject is completely free to choose its decision logic. Due to access details and further knowledge of the current tasks being executed, the decision happens on a more exact level than other models with a priori rights assignments can reach. At the end of the decision making process the authorizing subject must either authorize the operation (with all, e.g. 1/2, or only a part of its own rights, e.g., 1/3 instead of 1/2) or reject authorization. Rejection prevents the operation to be executed. A conditional authorization is used to include further subjects in the authorization process.

As only authorization and thus access control is directly involved for the extension of the existing model, it is not necessary to completely redefine the execution model from the perspective of the accessing subject. The main control flow of the accessing subject remains the same. Handling of multiple authorization happens through callbacks in a new separate extension.

This leads to a number of interfaces that an architecture integrating multiple authorization must provide or modify to meet the new needs. There are basically four interfaces provided by the security system:

- the security management interface to configure the access rights and define the conditions when an authorization is sufficient for an operation,
- the access control interface to check authorization of a particular operation and delaying its execution if it is subject to multiple authorization,
- the inspection interface to request details about the operation to be authorized for the authorizing subject (this interface itself must be secured from unauthorized access), and
- the authorization interface to finally grant or reject authorization for a particular operation.

Moreover, the participating subjects must provide two interfaces used for communication:

- the callback interface of the accessing subject called by the reference monitor if authorization is incomplete, and
- the authorization request interface of the authorizing subject called by the accessing subject to request authorization for the deferred operation call.

3. DISCUSSION & RELATED WORK

The presented architecture focuses on the practical integration on the base of existing security systems which often use rather simple DAC models. In this sense, it is not considered as a replacement of other more sophisticated security models which, however, also require a certain level of complexity in respect to the implementation of the reference monitor, the supporting infrastructure and the management of the whole system. In contrast, the presented concept with fractional rights is still sim-

ple to understand, requires less complex changes to the reference monitor leading to more robust implementations, and has a straight-forward integration of the existing DAC model. For example, for an integration in Java [2], it is necessary to extend the Policy class to support new fractional rights beside the old grant rules. A new SecurityManager must then be able to check for the fractional rights and do the callbacks if necessary as well as provide the interfaces for inspection and authorization as outlined above. A prototype for the Java integration is being developed to demonstrate the feasibility and possible applications.

The basic idea of involving several people in critical operations is not new. Various access control models have concepts to deal with separation of duty (e.g., RBAC [3], Policies [4]). With separation of duty two operations calls, e.g. on a particular object, must be performed by distinct subjects. This can then be used to increase security when, for example, someone who writes an order cannot approve an order and vice-versa. This, however, requires two different operations to exist for this purpose, i.e., a write method and an approve method on an order object. This assumption is not valid in many existing systems, e.g., in file systems that usually do not offer something like an approve read method on a file. (The presented architecture provides this split operations, thus could be used for this purpose.)

Other approaches using multiple credentials which are passed to the accessing subject are also possible. Here, though, the authorizing subjects loose direct control of their credentials once they have been given away. In addition, it requires them to fully identify all possible constraints they can put on the credential to limit the extent of the right they give away. In the end, this often means, that they have to foresee the exact purpose of the credential. With multiple authorization they authorize an actually happening operation call with ‘real live’ parameters and their final decision can even be based on some manual, in-person inspection, if in doubt.

Acknowledgement. The author would like to thank the members of the Munich Network Management (MNM) Team for helpful discussions and valuable comments on drafts of this paper. The MNM Team directed by Prof. Dr. Heinz-Gerd Hegering is a group of researchers at the University of Munich, the Munich University of Technology, and the Leibniz Supercomputing Center of the Bavarian Academy of Sciences.

Its webserver is located at http://wwwmnmteam.informatik.uni-muenchen.de/

REFERENCES

[1] M. Harrison, W. Ruzzo, J. Ullman. Protection in Operating Systems. In: *Communications of the ACM*, 19(8):461–471, August 1976.

[2] S. Oaks. *Java Security.* O’Reilly & Associates, 2nd edition, May 2001.

[3] R. Sandhu, E. Coyne, H. Feinstein, C. Youman. Role-Based Access Control Models. In: *IEEE Computer*, 29(2):38–47, February 1996.

[4] M. Sloman, E. Lupu. Security and Management Policy Specification. In: *IEEE Network*, 16(2):10–19, March/April 2002.