# Handling Capabilities in Security Policies

Salem Benferhat, Mouslim Tolba, Karim Tabia, Abdelkader Belkhir

# Handling capabilities in security policies

(Preprint version)

Salem Benferhat
CRIL
Univ. Artois & CNRS
F-62300 Lens
France

Mouslim Tolba
USTHB University
Algiers, Algeria

Karim Tabia
CRIL
Univ. Artois & CNRS
F-62300 Lens
France

Abdelkader Belkhir
USTHB University
Algiers, Algeria

## ABSTRACT

In existing access control systems, it is assumed that access control authorisation rules are defined on elementary actions and over concrete objects. This assumption does not hold in general. This paper proposes a slight extension of access control models where both elementary and non-elementary actions can be represented. A non-elementary action, called a plan, is a sequence of elementary actions, to be applied on objects, in order to achieve some task. We propose to represent a plan, denoted by P, as a partial pre-order over a subset of A x O where A is a set of elementary actions and O is a set of objects. We show how to derive explicit prohibitions in the presence of authorisation rules over plans.

## Keywords

Access control; Generic plans, sequences of actions.

## 1. INTRODUCTION

Access Control plays a crucial role in securing information or physical systems. For instance, to secure different areas of an airport, it is important to represent access control security policies that provide the list of authorisations /privileges associated with each user (passengers, grounded hostess, etc) of the airport.

In the literature, a large number of access control models have been proposed. The Lampson model [3] introduced matrix access control where a user gets a permission to run some action on an object, only if such action is stated in the matrix access. HRU [4] also uses an access matrix and was designed to improve the Lampson model to update security policies using the concept of commands. Discretionary Access Control(DAC)[3] models allow the handling of access rights to each object is in the absolute discretion of the owner or the subject who is responsible. This model is usually implemented with ACL (Access Control Lists). A simple example of DAC model is the one used by UNIX operating system. DAC models are not appropriate for airport security applications. In particular, they do not satisfy the least privilege principle [5].

RBAC (Role Based Access Control) [1] defines an authorisation model for accessing resources. Its main objective is to compactly describe security policies by introducing the concept of role which is a factorization of assigning permissions in order to prevent more relationships between permissions and users. Authorisation (or permissions) are not directly assigned to user but to their abstraction called role.

Existing access control models consider that actions executed by users are elementary. however, in practice security policies are more defined over sequences of actions rather then over elementary actions.

For instance, in an airport security example, a security policy may concern a simple activity such as checking a luggage of a given passenger. The security policy may also contain authorisation rules that deal with complex activities such as the whole boarding process. This complex activity (boarding process) is in fact a sequence of elementary actions: checking luggage, assigning seats, printing boarding pass,... Another example where both elementary actions and non-elementary actions co-exist is the one of Linux operating system. In Linux systems, the actions (or commands) "ls","grep","type" are high-level actions that are implemented by set of low level actions(system calls).

In [2], a slight extension of OrBAC (Organisation based access control) model to integrate non-elementary actions[7] [5][6]. Non elementary actions are represented by a partial pre order overs pairs $(a_i,o_j)$ where $a_i$ is an elementary action and $o_j$ is a concrete object. These non-elementary actions will be simply called plans.

This paper goes one step further by showing how explicit prohibitions can be represented over non elementary actions,which is detailed in next section.

## 2. REPRESENTING SEQUENCE OF ACTIONS

In existing access control systems, including the OrBAC model described above, it is assumed that authorisation rules are defined on elementary actions and over concrete objects. This assumption does not hold in general.

In [2] a slight extension of OrBAC model called Pl-OrBAC has been proposed. In Pl-OrBAC Both elementary and non-elementary actions can be represented. The main idea is to modify the abstract entity "permission" and the concrete entity "is permitted" in order to deal with non-elementary activities and non-elementary actions respectively. This paper goes one step further and show how to represent and derive

concrete prohibitions. Let us first give a brief refresher on OrBAC model.

## 2.1 OrBAC model

Organization-based access control (OrBAC) [7] [6] is an access control model that allows the policy designer to define a security policy in a compact way. OrBAC model is based on the following four principles:

Organization: An organisation may be an institution or an organized group of subjects, playing some role.

Levels of abstraction: A concrete level materialized through the use of three concrete entities: subjects, actions and objects.

Relating abstract authorisation to concrete authorisation: subjects are abstracted into roles. Similarly, an activity is a set of actions to which the same security rule applies. And a view is a set of objects to which the same security rule applies.

Figure 1 shows the interactions between existing entities in the model OrBAC:

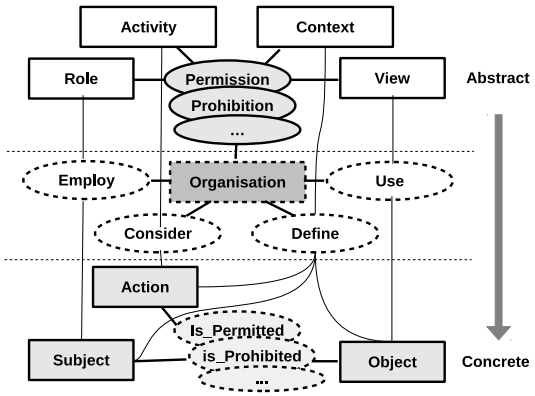


**Figure 1: OrBAC model**

## 2.2 Representing explicit prohibitions over sequences of actions

This section shows how to represent and derive explicit prohibitions over non-elementary actions in Pl-OrBAC model. A non-elementary activity, called a generic plan[2], is a sequence of elementary activities, to be applied on views, in order to achieve some task. Let $G_A$ be a set of activities and $G_V$ be a set of views. A generic plan, denoted by the capital letter Pl, is a POS (partially pre-ordered set) Pl=(NE,$\prec$) where NE$\subseteq G_A$x$G_V$ and $\prec$ is partial pre-order on NE.

The partial pre-order indicates the precedence order between the activities.

In order to take into account the prohibition relations over generic plans, we replace the prohibition and permission relations in OrBAC by new relations called G-prohibition and G-permission. Generic plans (Figure 2) is a new entity that contains generic plans.

At the concrete level, we also need to introduce concrete plans. Concrete plans are directly defined over concrete actions and concrete objects rather then on their abstractions. More precisely let CA be a set of concrete actions, O be a set objects. A concrete plan, denoted by the lower case p, is totally-ordered set p=(E, $\prec$) where E$\subseteq$CA x O and $\prec$ is a total order on E.

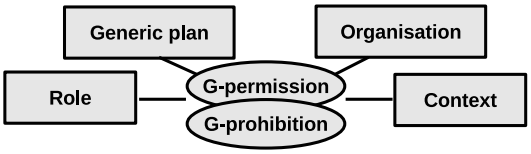


**Figure 2: Representation of generic plans, G-permission and G-prohibition relations**

In order to take into account the concept of concrete plans, we also need to introduce the new entity, called concrete plan. Concrete permissions and concrete prohibitions, denoted G_ispermitted and G_isprohibited, is described by Figure 3:

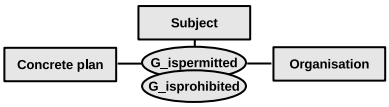


**Figure 3: Concrete plans, G-ispermitted and G-isprohibited relations.**

In [2] an algorithm has been proposed to derive permissions overs concrete plans. Algorithm 1 shows how to derive concrete prohibitions attributed to a particular user u to achieve a particular plan p. Namely, how to derive G_isprohibited(org,u,p).

**Algorithm 1:** Derivation of concrete explicit prohibitions

**Input :** u : user
p' : a concrete plan
Pl-OrBAC : model
**Output:** Yes/no there is an explicit prohibition to the uses user u to achieve p

```
begin
    forall Subplans p' of P do
        Let P={G-prohibition(org,r,Pl,c) such that:
         G-prohibition(org,r,Pl,c) is an element of
         G-prohibition table and p' is an instantiation
         of Pl}
        foreach G − prohibition(org, r, Pl, c) in P do
            if (Empoyer(org, u, r) is an element of
             empoyer relation and
             define(org, u, p') is an element of Define
             relation) then
                return true
            end
        end
        return false
    end
end
```

The main idea of this algorithm is to see whether there exists a generic G-prohibition rule that applies for the concrete user u and any concrete subplan p' of p. Hence, the idea is to generate all prohibitions $G_P$ on generic plans Pl that accept p' as an instantiated plan. An instantiated plan is a concrete plan obtained by replacing roles, activities and

views by concrete users, actions and objects respectively. Note that there is an explicit prohibition for a plan p if one can derive an explicit prohibition for each subplan p' of p.

# 3. EXAMPLE

Assume that we are interested in modeling access control security rules concerning the boarding process of a given flight. To achieve this task, we need to specify the content of each table and relation of the Pl-OrBAC model.

First, we assume that we only have one organisation, which is "HB", Simply denoted by HB(see Figure 4)


**Figure 4: The content of organisation table.**

Now, we need to specify the list of roles, activities and views used in the information system of the organisation HB:

- *Roles:* We assume that there are three roles: passengers, ground hostess, ground and security hostess.
- *Activities:* The activities in HB are: checking luggages, assigning seats and boarding passengers.
- *Views:* We assume that the list of views is: seats, luggages to check and passengers to board.

The second step is to identify the list of concrete objects in the information system, the list of concrete actions on the information system and the list of concrete users:

- *Users:* John, Bob, Alice and Pierre.
- *Actions:* there are five actions: Print luggage tags at air company checking counter, print luggage tags at air company checking kiosks, assign a seat at air company checking counter, assign a seat at air company checking kiosk and use kiosk1 to validate boarding pass.
- *Objects:* Seat 1 to Seat 400, L1, L2, John, Bob.

Figure 5 describes the relation "Empower" that relates users to roles.

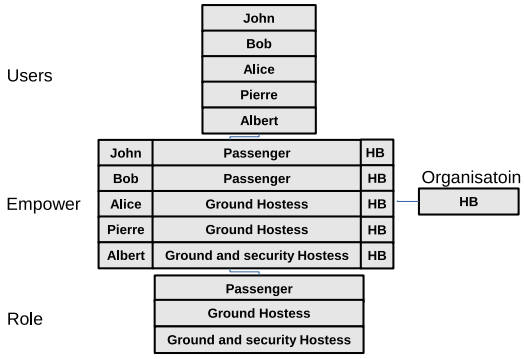


**Figure 5: The relation "Empower"**

Figure 6 gives the content of the relation "consider" that relates concrete actions used in the information system to general activities:

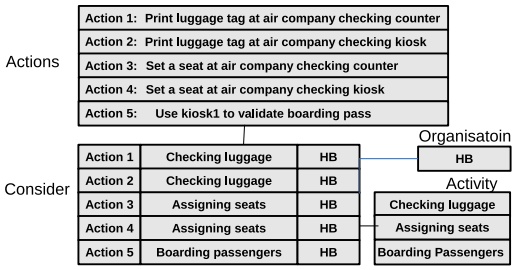


**Figure 6: The relation "consider"**

Figure 7 gives the content of the relation "Use" that relates objects of the information system to views. For instance, the object Seat1 is used as the view Seat, and the concrete object L1 is used as a luggage to check.

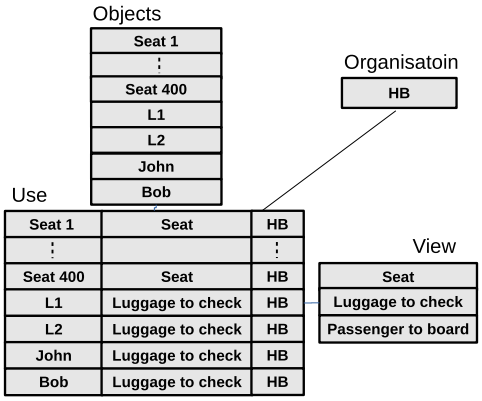


**Figure 7: The relation "use"**

Figures 8 gives the table "context" and its definition relation. In this example, we assume that there are only three contexts, denoted by "flight opened for check in", "flight opened for boarding" and "always". Figures 9 gives the con-


**Figure 8: The context table**

text table.

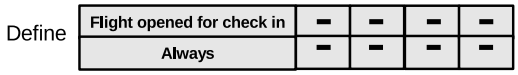


**Figure 9: The relation "define"**

The context "always" simply means that no condition is required for achieving some authorisation rules. While the context "Flight open check in" states that authorisation rules only apply if the flight is open for check in. In our example, from the relation "define" the context "flight open for check in". However, the context "flight open for boarding" is false. This means that all authorisation rules that need the context "flight open for boarding" cannot be activated.

The set of authorisations/ prohibitions security rules used by the HB organisation as follows:

- "Grounded hostess" has permission to assign seats without additional condition. Said differently the context of this permission rule is "always".
- "Grounded hostess" has permission to check luggage when a flight is open for check in (namely, in the context flight open for check in)
- It is prohibited for "grounded hostess" to achieve the activity of boarding passengers.

We assume that we have two plans (sequences of elementary activities):

- P1: This generic plan is defined by: ($NE_1$={(checking luggage, luggage), (assigning seats, seats)}, $\prec_{NE_1}$,= $\emptyset$)
- P2: This generic plan is defined by: $NE_1$={(checking luggage, luggage), (assigning seats, seats),(boarding passengers, passengers)} with (checking luggage, luggage) $\prec_{NE_2}$ (boarding passengers, passengers) and (assigning seats, seats) $\prec_{NE_2}$ (boarding passengers, passengers).

Intuitively, P1 corresponds to the complex activity of check in. while P2 involves the whole boarding process (check in luggages and boarding passengers). $\prec_{NE_1}$ states that there is no precedence relation between the two elementary activities: check in luggages and assigning seats. $\prec_{NE_2}$ states that check in luggages (resp assigning seats) should be done before boarding passengers. Given these two generic plans, we have two additional access control security rules:

- Grounded Hostess have permission to achieve P1 in the context of "flight open for check in".
- Security hostess have permission to achieve P2 in the context of flight open for boarding.
- It is prohibited for grounded hostess to achieve P2.

Figure 10 and 11 give the relations G-permission and G-prohibition :

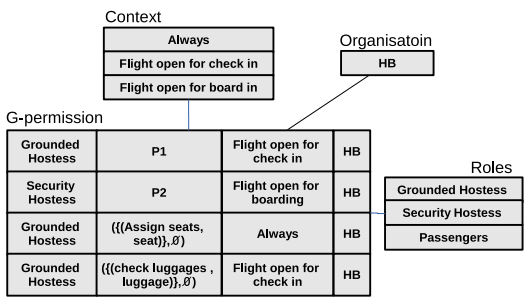


**Figure 10: G-permission relation**

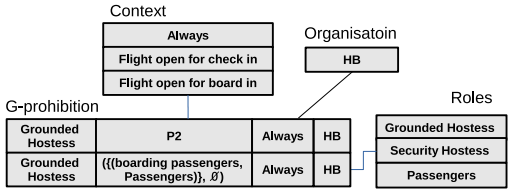


**Figure 11: G-prohibition relation**

# 4. CONCLUSIONS

This paper shows how to represent prohibition rules over non-elementary actions in access control model. We provide a slight extension of OrBAC model and show how to integrate the concepts of generic and concrete plans by means of sequences of actions. A future work is to deal with the problem of handling conflicts when security policies, that use different levels of granularity of actions, co-exists.

## Acknowledgements

This work has received support from the french national project PEPS SISC INS2I 2016 entitled EPIAGE (Evolution des politiques de contrôle d'accès et gestion d'actions interférentes).

# 5. REFERENCES

[1] Muhammad Umar Aftab, Muhammad Asif Habib, Nasir Mehmood, Mubeen Aslam, and Muhammad Irfan. Attributed role based access control model. In *2015 Conference on Information Assurance and Cyber Security (CIACS)*, pages 83–89. IEEE, 2015.

[2] Salem Benferhat, Mouslim Tolba, Karim Tabia, and Abdelkader belkhir. Integrating non elementary actions in access control models. In *To appear in proceedings of 9th International Conference on Security of Information and Networks*, 2016.

[3] David F Ferraiolo, Ravi Sandhu, Serban Gavrila, D Richard Kuhn, and Ramaswamy Chandramouli. Proposed nist standard for role-based access control. *ACM Transactions on Information and System Security (TISSEC)*, 4(3):224–274, 2001.

[4] Michael A Harrison, Walter L Ruzzo, and Jeffrey D Ullman. Protection in operating systems. *Communications of the ACM*, 19(8):461–471, 1976.

[5] Lihui Hu, Jean Mayo, and Charles Wallace. An empirical study of three access control systems. In *Proceedings of the 6th International Conference on Security of Information and Networks*, pages 287–291. ACM, 2013.

[6] Aziz Kaddani, Amine Baina, and Loubna Echabbi. Towards a model driven security for critical infrastructures using orbac. In *Multimedia Computing and Systems (ICMCS), 2014 International Conference on*, pages 1235–1240. IEEE, 2014.

[7] Anas Abou El Kalam, RE Baida, Philippe Balbiani, Salem Benferhat, Frédéric Cuppens, Yves Deswarte, Alexandre Miege, Claire Saurel, and Gilles Trouessin. Organization based access control. In *Policies for Distributed Systems and Networks, 2003. Proceedings. POLICY 2003. IEEE 4th International Workshop on*, pages 120–131. IEEE, 2003.