

# O2O: Virtual Private Organizations to Manage Security Policy Interoperability

Frédéric Cuppens, Nora Cuppens-Boulahia, and Céline Coma

GET/ENST Bretagne, 2 rue de la chataîgneraie,  
35576 Cesson-Sévigné Cedex, France  
{frederic.cuppens,nora.cuppens,celine.coma}@enst-bretagne.fr

**Abstract.** Nowadays, the interaction between systems is absolutely essential to achieve business continuity. There is a need to exchange and share services and resources. Unfortunately, this does not come without security problems. The organizations (companies, enterprizes, etc.) have to manage accesses to their services and resources by external opponents. O2O is a formal approach we suggest in this paper to deal with access control in an interoperability context. It is based on two main concepts: *Virtual Private Organization* (VPO) and Role Single-Sign On (RSSO). A VPO enables any organization undertaking an inter-operation with other organizations to keep control over the resources accessed during the interoperability phases. The RSSO principle allows a given subject to keep the same role when accessing to another organization but with privileges defined in the VPO. Thus, using O2O, each organization can define and enforce its own secure interoperability policy. O2O is integrated in the OrBAC model (Organization based access control).

**keywords:** Virtual Organization (VO), Virtual Private Organization (VPO), Role Single Sign On (RSSO), OrBAC, Access Control, Authority spheres, Interoperability

## 1 Introduction

Current information systems are more and more distributed and require more interactions with external components or services. Classical client/server architecture is no longer adapted to manage these systems when several organizations have to exchange information or provide service interoperability. For this purpose, the concept of *Virtual Organization* (VO) has been suggested [13,18]. A VO is created by several organizations and is formed with some users, services or resources from the different organizations.

To interact securely in a VO, each organization has to define its security policy and every interaction must be compliant with the policies of organizations involved in the interaction. Classical security models such as the RBAC model [19] are not suited to model these requirements. They only apply to centralized management of security in which some servers implement their security policy to control the access of some clients. However, in a VO, each principal wants to have some guarantee that information provided to other principals are processed securely. Thus, every principal implements its own policy and the

interaction will only take place if it does not violate their policy. Access control models have to be adapted to manage such symmetric behavior.

In this paper, we suggest a new approach called O2O (for *Organization to Organization*) to manage interoperability between components having their own policies defined by different organizations. To explain the basic principals of O2O, let us consider that a given organization Alice.org wants to interoperate with another organization Bob.org. In this case, each organization has to define a *Virtual Private organization* (VPO) respectively called Alice2Bob (A2B for short) and Bob2Alice (B2A for short). The VPO A2B is associated with a security policy that manages how subjects from organization Alice.org may have an access to organization Bob.org. We say that the VPO A2B manages the *interoperability* security policy from Alice.org to Bob.org. The VPO B2A is similarly defined to control accesses of subjects from Bob.org to Alice.org. Hence, a VPO is a dynamic organization created to achieve a given purpose of interoperability and disappears once this purpose in no more needed. VPO is the basic concept used to enforce our O2O approach.

O2O is formally defined as an extension of the OrBAC model [1,8]. The concept of organization is central in this model. In OrBAC, each organization can define its own security policy which does not directly apply to concrete subjects, actions and objects. Instead, the policy is defined using the organizational concepts of role, activity and view which are respectively abstractions of subject, action and object (this is further detailed in section 3 below). The concrete policy that applies to subjects, actions and objects is then derived from the organizational policy specification.

We suggest extending the OrBAC model so that an organization can also define interoperability security policies with other organizations using O2O. One advantage of the approach is that a given subject who is assigned to a given role in an organization A can keep his or her role when accessing to another organization B but possibly with different privileges as defined in the interoperability policy of A2B. By analogy with Single-Sign On used for identity management, we call this principle RSSO for Role Single-Sign On. For instance, let us assume that a given subject is assigned to role physician in a given hospital A. Then, when this subject attempts to have an access to another hospital B, this subject is assigned to role physician, not in hospital B, but in the VPO A2B. By doing so, this subject can keep his or her role but possibly with different privileges than other subjects that are directly assigned to role physician in B. Notice that privileges of role physician in the virtual private organization A2B are also generally different from those of role physician in organization A.

The remainder of this paper is organized as follows. Section 2 further motivates our approach and compares it with other proposals. Section 3 gives an overview of the OrBAC model. Section 4 defines the confinement principle. We argue that this principle must be enforced so that secure interactions between organizations may take place. Section 5 presents our O2O approach and shows how it is used to manage security in a virtual organization. We formally define inter-organization compatibility in section 6 which makes it easy interactions between organizations that need to interoperate. Section 7 shows how our O2O model applies to manage security of a Virtual Organization. Finally, section 8 concludes the paper.

## 2 Motivation

Federation of identity currently facilitates the subject's authentication through the Internet. For example, using Liberty Alliance [4], some subject has not to repeatedly identify himself or herself with a new password each time he or she wants to have an access to a service managed by a different organization. Hence, this subject only manages a single password (single-sign on [15]) and then, through certificate exchanges, he or she can access to other organizations that belong to the Alliance without the halting delays of redundant entry.

However, at first glance, except the real advantage that the authentication is done once for all, this process is not fully satisfactory as it is based on a high level consideration of Alliance membership. Hence, if A and B are two members of the alliance then (1) all authenticated subjects of B will have the same set of rights when accessing to A and (2) the access control is not fine grained as it is based on some characteristics shared by all B's subjects vis-à-vis of A. For instance, let us consider a subject John who is assigned to role customer into a first organization A and that another organization B gives a special discount to the customers of organization A. Once John is logged in organization A, if both A and B are members of Liberty Alliance, then B will trust in the certificate delivered by the Alliance that says that John comes from organization A. However, this certificate does not guarantee that John is assigned to role customer in organization A.

This need for finer grained dynamic access control has been identified by Liberty Alliance and is included in the SAML 2.0 specification. The solution is based on exchange of *credentials* [12]. A credential contains attribute information used to establish trust between two parties A and B. Traditionally, the credential is sent as a whole. In this way, though not necessary, all information contained in the credential is disclosed. To avoid this disclosure, the other drastic measure is to reveal nothing about the information contained in the credential. In these traditional approaches, interaction between some parties A and B belonging to some Alliance can fail though for both A and B the amount of information that has to be disclosed is acceptable.

Some works have been done to manage a more subtle access control policy to the information contained in the credentials and to define classes of negotiation strategies to control disclosure of credentials during the interaction. TrustBuilder [10,22] provides a set of language-independent protocols that ensures the interoperability of the negotiation strategies. During a negotiation, a local security agent uses a negotiation strategy to determine which local resources to disclose next and to accept new disclosure from other participants. Trust-X [3] also supports different strategies for trust negotiations to determine the order in which credentials should be disclosed.

Besides the definition of negotiation strategies, we also need high level languages to enable parties who intend to inter-operate to specify authorization requirements that must be met to access to all or part of the information contained in the credential. An example of such a language is presented in the work by J. Li, N. Li and W. Winsborough [11]. Though this language that defines security policies based on credential are inspired by the well known access control model RBAC, it is not free of ambiguity. There is no clear separation between the security policy specification and credential that are used

to implement this policy. A similar comment applies to other approaches such as the Dynamic Coalition-Based Access Control (DCBAC) model [21].

Moreover, as we said in the introduction, the use of RBAC as access control model is not judicious. As a matter of fact, the role approach leads to a more fine grained access control to attribute information transmitted in the credential than the Liberty Alliance or Micro Passport [14] access control do. But the interoperability enabled by such a role approach is static; all the roles needed for this interaction must be anticipated and have to be defined in advance.

We actually need approaches that allow organizations to specify more dynamic access control policies. This is typically the case when several organizations interoperate through a Virtual Organization. Grid computing is especially concerned with the sharing and coordinated use of resources in such distributed VOs [9]. In a VO, one must guarantee interoperability of diverse local mechanisms, support dynamic creation of services and enable dynamic creation of trust domains. For instance, the CAS server [16] allows VOs to express policies and communicates these policies to various local organizations involved in the VO. Thus, CAS implicitly assumes a centralized administration of the VO security policy and does not answer the following questions [17]: (1) Who is in charge of defining the security policy associated with the VO, (2) Who is responsible for administering this security policy?

Since there are several organizations involved in the VO having possibly conflicting interests, it is not easy to answer these two questions if we assume a centralized administration of the VO security policy. This is why we argue for a decentralized administration of the VO and we suggest our O2O approach whose objectives are the following:

- In O2O, each organization involved in a VO is responsible for controlling interoperability with other organizations through the definition and administration of a Virtual Private Organization (VPO). VPO provides a fully decentralized administration of a VO.
- In a VPO, it is not necessary to define new roles. When accessing to another organization, a given subject can keep the same role as in its own organization, but the VPO will define the security policy associated with this role in this other organization. Thus, in some sense, VPO generalizes the concept of single sign on already defined for subject identity to *role* identity. This is why we call this principle RSSO (Role Single-Sign On).
- In a VPO, it is possible to specify dynamic and fine grained access control. In particular, the *organization* concept as it is defined in the OrBAC model brings the *dynamical* dimension through our O2O approach.
- Our O2O model clearly separates the security policy specification from its implementation. In this paper, we only present the policy specification part of our model. We do not address its implementation through trust negotiation by exchange of credentials.

### 3 Outline of the OrBAC model

Since O2O is actually defined as an extension of the OrBAC model, we first briefly present the basic principles of OrBAC.

The concept of *organization* is central in OrBAC. Intuitively, an organization is any entity that is responsible for managing a security policy. Each organization can use OrBAC to specify its own security policy at the *organizational* level, that is abstractly from the implementation of this policy. Thus, instead of modelling the policy by using the concrete and implementation-related concepts of subject, action and object, the OrBAC model suggests reasoning with the roles that subjects, actions or objects are assigned in the organization. The role of a subject is simply called a *role* as in the RBAC model. On the other hand, the role of an action is called an *activity* whereas the role of an object is called a *view*.

Each organization can then define security rules which specify that some roles are permitted or prohibited to carry out some activities on some views. These security rules do not apply statically but their activation may depend on contextual conditions. For this purpose, the concept of *context* is explicitly introduced in OrBAC. Thus, using a formalism based on first order logic, security rules are modelled using a 6-places predicate:

- `security_rule(type, org, role, activity, view, context)` where `type` belongs to {`permission`, `prohibition`}.

For instance, the following security rule:

- `security_rule(permission, a_hosp, nurse, consult, medical_record, urgency)` means that, in organization `a_hosp`, a nurse is permitted to consult a medical record in the context of `urgency`.

The organizational policy is then used to automatically derive concrete configurations of PEP's (Policy Enforcement Point, see the AAA architecture). For this purpose, we need to assign to subjects, actions and objects, the roles they are assigned in the organization. In the OrBAC model, this is modelled using the following 3-places predicates:

- `empower(org, subject, role)`: means that in organization `org`, `subject` is empowered in `role`.
- `consider(org, action, activity)`: means that in organization `org`, `action` is considered an implementation of `activity`.
- `use(org, object, view)`: means that in organization `org`, `object` is used in `view`.

For instance, the fact `empower(a_hosp, john, physician)` means that organization `a_hosp` empowers `john` in role `physician`.

Notice that, instead of enumerating facts corresponding to instances of predicate `empower`, it is also possible to specify *role definitions* which correspond to logical conditions that, when satisfied, are used to derive that some subjects are automatically empowered in the role associated with the role definition. Role definition may be viewed as an extension of the Attribute-Based User-Role assignment suggested in [2]. For instance, a bookshop `bs` may consider that a subject is empowered in role `gold_customer` if this subject is empowered in role `customer` and if he or she has been a customer for more than ten years. This is modelled by the following role definition:

- `empower(bs, X, gold_customer) :-  
empower(bs, X, customer), membership(bs,X,Y), Y >= 10.`

Activity and view definitions are similarly used to automatically manage assignment of action to activity and object to view. Role definition (resp. activity and view definition) are used in our O2O approach for managing federations of roles (resp. activity and view) as specified in interoperability security policy (see section 5).

Regarding context, we have also to define logical conditions to characterize when contexts are active. In the OrBAC model, this is represented by logical rules that derive the following predicate:

- `hold(org, subject, action, object, context):` means that in organization `org`, subject performs action on object in context `context`.

Using the model, one can then derive concrete privileges that apply to subject, action and object from organizational security rules. This corresponds to the following general principle of derivation:

- `concrete_privilege(Type,S,Act,Obj) :-  
security_rule(Org,Type,R,A,V,Cxt),  
empower(Org,S,R) use(Org,Obj,V), consider(Org,Act,A),  
hold(Org,S,Act,Obj,Cxt).`

that is a subject `S` has a concrete privilege of type `Type` to perform an action `Act` on object `O` if (1) organization `Org` assigns to role `R` a security rule of type `Type` to perform activity `A` on view `V` in context `Cxt` and (2) if `Org` empowers subject `S` in role `R` and (3) `Org` uses object `O` in view `V` and (4) `Org` considers that action `Act` implements the activity `A` and (5) in `Org`, the context `Cxt` is active when subject `S` is performing action `Act` on object `Obj`.

This general principle of derivation of concrete privileges from organizational authorizations is used to automatically generate concrete configurations (see [5] for further details in the case of network security policies). We actually restrict our model to be compatible with a stratified Datalog program [20], so that derivation in our model is computable in polynomial time.

## 4 Confinement principle

Since OrBAC can explicitly manage several security policies related to different organizations, it provides an adequate framework to define security policies for organizations that have to interoperate. For instance, let us consider two hospitals `a_hosp` and `b_hosp`. We assume that each hospital manages its own medical records into a view `medical_record` and that there is no medical record shared by the two hospitals. We also assume that each organization manages a role `physician` and that a physician is permitted to consult medical records handled by his or her hospital in every situation (corresponding to a context called `nominal`). In OrBAC, this is modelled by the following security rule:

- `security_rule(permission, a_hosp, physician, consult, medical_record, nominal)`. and similarly for `b_hosp`.

Up to now, notice that a physician from a given hospital cannot have an access to medical records managed by the other hospital. More precisely, OrBAC actually implements a *confinement* principle so that the scope of every security rule is restricted to the organization in which the rule applies. Thus the rule above specifies that a physician is permitted to consult medical records of `a_hosp` but prevents him to have an access to medical records of `b_hosp` (except if some medical records are physically shared by both `a_hosp` and `b_hosp` but this is not our assumption here).

Confinement is a good security principle. When dealing with interoperability requirements, we have to be compatible with this principle. For instance, let us assume that, in a context of urgency, `a_hosp` wants to grant to physician from `b_hosp` the permission to consult its medical records. A first possibility, which preserves the confinement principle, would be that `a_hosp` creates a new role `b_physician` and specifies the following security rule:

- `security_rule(permission, a_hosp, b_physician, consult, medical_record, urgency)`.

and the following role definition:

- `empower(a_hosp, X, b_physician) :- empower(b_hosp, X, physician)`.

This role definition rule says that a subject `X` is empowered in role `b_physician` in organization `a_hosp` if this subject is empowered in role `physician` in organization `b_hosp`.

So let us consider a subject Alice who is empowered in role `physician` in organization `b_hosp` and that context `urgency` is active in organization `a_hosp`. If Alice asks for an access to an `a_hosp`'s medical record, then she will have to give her credential that proves that she is empowered in role `physician` in organization `b_hosp`. By doing so, she will automatically be empowered in role `b_physician` in organization `a_hosp` and thus will be permitted to have an access to `a_hosp`'s medical records since we assumed that the context `urgency` is active.

However, this first solution to manage interoperability is not satisfactory for at least three reasons:

1. From the security administration point of view, it will create many “artificial” roles related to interoperability, such as the `b_physician` role. At the end, this may significantly complicate administration of the security policy.
2. From the usability point of view, physicians from organization `b_hosp` will have to manage several different roles: role `physician` in organization `b_hosp`, role `b_physician` in organization `a_hosp`, etc. This is not satisfactory.
3. From the interoperability point of view, physicians from organization `b_hosp` may be *temporarily* permitted to have an access to `a_hosp` (for instance they are asked to intervene after some catastrophic event). When the need for interoperability ends, it is necessary to revoke physicians from their role `b_physician`. This is not convenient and a more dynamic approach is necessary.

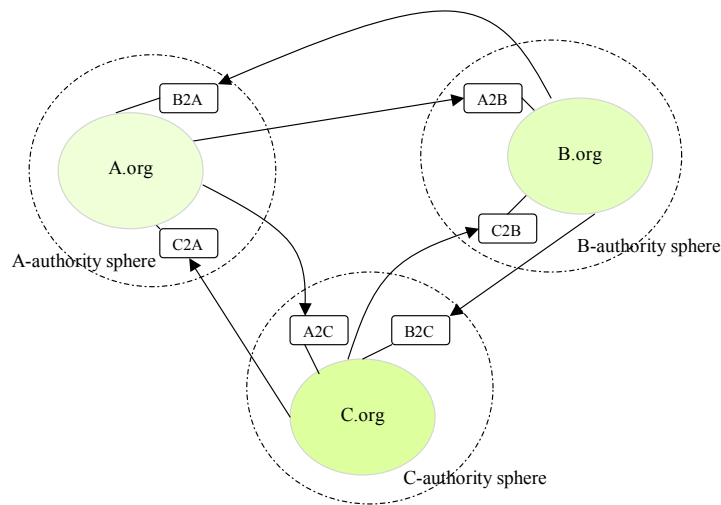
Thus, one of the objectives of our O2O approach is to provide a more adequate management of role (and also of activity and view).

## 5 O2O and Virtual Private Organization

In some sense, our O2O model is similar to management of identity provided by federations of identity such as Liberty Alliance. When a given organization A wants to interoperate with another organization B, it has first to create a Virtual Private Organization (VPO) B2A and associate B2A with a security policy to control interoperability from B to A (see figure 1 for an application of the O2O approach with three organizations). Similarly organization B will create another VPO A2B to control interoperability from A to B.

Thus, each VPO is associated with two attributes: o-grantor and o-grantee. The o-grantor attribute represents the organization which has created the VPO in order to grant some accesses to subjects coming from another organization represented by the o-grantee attribute.

As shown in figure 1, a VPO is within the *sphere of authority* [7] of the o-grantor attribute. A sphere of authority defines an oriented relation between organizations whose meaning is the following: An organization A is in the sphere of authority of another organization B if the security policy that applies to A is defined and administrated by B. This administration is done thanks to the AdOrBAC model principles (see [6] for more details).



**Fig. 1.** Security policy interoperability between three organizations

Thus, in a VPO, the o-grantor organization can define roles, activities and views and associate these roles, activities and views with contextual security rules as in a classical

organization of the OrBAC model. When assigning subjects, actions and objects to the respective roles, activities and views defined in the VPO, the following restrictions apply:

- If a given role is assigned to a given subject in a VPO, then this subject must come from the o-grantee organization of this VPO. This is because a VPO is designed to control how subjects from the o-grantee organization may have an access to the o-grantor organization.
- If a given object is used in a given view in a VPO, then this object must be used in some view in the o-grantor organization. This is because, in a VPO, the o-grantor organization can only grant access to its “own” objects.
- If a given action is considered an implementation of an activity, then this action must be controlled by the o-grantor organization. This will be the case when this action is directly implemented by the o-grantor organization (for instance, a web service provided by the o-grantor). When this action actually comes from the o-grantee organization (for instance a mobile code such as an applet), the situation is more complex. In this case, the o-grantor organization will choose either to (1) not trust this code and to reject it or (2) execute it in a confined space or (3) ask some guarantee to the o-grantee organization. In this latter case, we consider that this must be part of the negotiation protocol.

In the above example, since a\_hosp wants to interoperate with b\_hosp, a VPO b\_hosp2a\_hosp is created (bh2ah for short in the following). To grant to physicians from b\_hosp the permission to consult medical records in organization a\_hosp in the context urgency, we have to add to the VPO bh2ah security policy the following security rule:

- security\_rule(permission, bh2ah, physician, consult, medical\_record, urgency).

and the following role definition:

- empower(bh2ah, X, physician) :- empower(b\_hosp, X, physician).

In particular, notice that physicians from b\_hosp will keep their role when accessing to a\_hosp (RSSO principle). Notice also that the O2O approach clearly separates the physician’s permissions in the VPO bh2ah from their permissions in both a\_hosp and b\_hosp.

This is a simple example. We can imagine more complex examples. For instance, in September, a bookshop bs may grant special discount on scientific books to students of university u who have a credit card and are more than 18 years old. This is modelled as follows:

- security\_rule(permission, u2bs, student, special\_discount, scientific\_book, september).
- empower(u2bs, X, student) :-  
empower(u, X, student), age(X,Y), Y >= 18, credit\_card\_holder(X).

Of course, implementation of the O2O is based on credential exchange. For instance, in the above example, a student will have to exchange her credential to prove she is actually a student from university u, is more than 18 years old and is a credit card holder. However, compared with previous approaches such as [11,21], one advantage of O2O is that it clearly separates the specification of the interoperability security policy from its implementation with credentials.

## 6 Security policies compatibility

One of the advantages of the OrBAC model is that the security policy specification of an organization is structured using the concepts of role, activity, view and context. To ease the definition of interoperability security policies, it is possible that some organizations will agree that some correspondances exist between their respective roles, activities, views and contexts.

Let us first present the concept of role compatibility. If a given organization B agrees that a given role `role_A` of another organization A is compatible with one of its role `role_B`, then every subject empowered in role `role_A` in organization A will automatically be granted the privileges of `role_B` when accessing to organization B.

To model role compatibility, we introduce the following predicate:

- `compatible_role(A2B,role_A,role_B)`: in the virtual organization A2B, organization B agrees with organization A that `role_A` defined in organization A is compatible with `role_B` defined in organization B.

Using this predicate, we can automatically derive some part of the interoperability policy by considering the following rule:

```
R1: security_rule(Type,A2B,Role_A,Activity,View,Context) :-  
    o-grantee(A2B,A), o-grantor(A2B,B),  
    security_rule(Type,B,Role_B,Activity,View,Context),  
    role_compatible(A2B,Role_A,Role_B).
```

This rule says that if organizations A and B agree that `Role_A` is compatible with `Role_B`, then every security rule assigned to `Role_B` in the organization B are also assigned to `Role_A` in the VPO A2B.

The approach based on view, activity and context compatibility is different. Thus, let us assume that a given organization B agrees to consider that a given view `view_A` (resp. activity `activity_A`) (resp. context `context_A`) of organization A is compatible with one of its view `view_B` (resp. activity `activity_B`) (resp. context `context_B`). Now, if a given role `role_A` of organization A is permitted to perform `activity_A` on `view_A` in context `context_A`, then this role `role_A` will be automatically permitted to perform `activity_B` on `view_B` in context `context_B` in organization B.

This is modelled by using three other predicates `compatible_view`, `compatible_activity`, and `compatible_context` whose meaning is similar to predicate `compatible_role`. We then consider the following rule which is used to automatically derive some part of the interoperability policy:

```
R2: security_rule(Type,A2B,Role,Activity_B,View_B,Context_B) :-  
    o-grantee(A2B,A), o-grantor(A2B,B),  
    security_rule(Type,A,Role,Activity_A,View_A,Context_A),  
    activity_compatible(A2B,Activity_A,Activity_B),  
    view_compatible(A2B,View_A,View_B),  
    context_compatible(A2B,Context_A,Context_B).
```

This rule says that if organizations A and B agree that `Activity_A`, `View_A` and `Context_A` are respectively compatible with `Activity_B`, `View_B` and `Context_B`, then security rules

that apply in the organization A also applies in the VPO A2B after replacing Activity\_A in Activity\_B, View\_A in View\_B and Context\_A in Context\_B.

To illustrate rules R1 and R2, let us consider two organizations french and nato that respectively correspond to French and Nato Defense Organizations. The policy of the nato organization includes the following security rules:

S1: security\_rule(permission,nato,nato\_confidential,read,nato\_confid\_doc,need\_to\_know).

S2: security\_rule(permission,nato, nato\_secret, read, nato\_secret\_doc, need\_to\_know).

Rule S1 says that in organization nato, subjects empowered in role nato\_confidential (correspond to subjects cleared at level nato\_confidential) are permitted to read documents in view nato\_confid\_doc (correspond to documents classified at level nato\_confidential) in context need\_to\_know. Rule S2 is similar to rule S1 but applies to subjects cleared at level nato\_secret that are permitted to read documents classified at level nato\_secret.

Let us now assume that organizations french and nato create two VPOs fr2nato and nato2fr to manage their interoperability and that they agree about the following compatibilities:

F1: role\_compatible(fr2nato, confidentiel\_defense, nato\_confidential).

F2: activity\_compatible(nato2fr, read, lire).

F3: view\_compatible(nato2fr, nato\_confidential\_doc, doc\_cd).

F4: view\_compatible(nato2fr, nato\_secret\_doc, doc\_cd\_special\_fr).

F5: context\_compatible(nato2fr, need\_to\_know, besoin\_de\_connaitre).

Then from R1, S1 and F1, we can derive the following security rule:

security\_rule(permission,fr2nato,confidentiel\_defense,read,nato\_confid\_doc,need\_to\_know).

Actually, from F1, we can derive that, when accessing to organization nato, subjects empowered in role confidentiel\_defense in organization fr will get the same permissions as subjects empowered in role nato\_confidential in organization nato.

From R2, S1, F2, F3 and F5, we can derive:

security\_rule(permission,nato2fr, nato\_confidential, lire, doc\_cd, besoin\_de\_connaitre).

And from R2, S2, F2, F4 and F5, we can derive:

security\_rule(permission,nato2fr,nato\_secret,lire, doc\_cd\_special\_fr, besoin\_de\_connaitre).

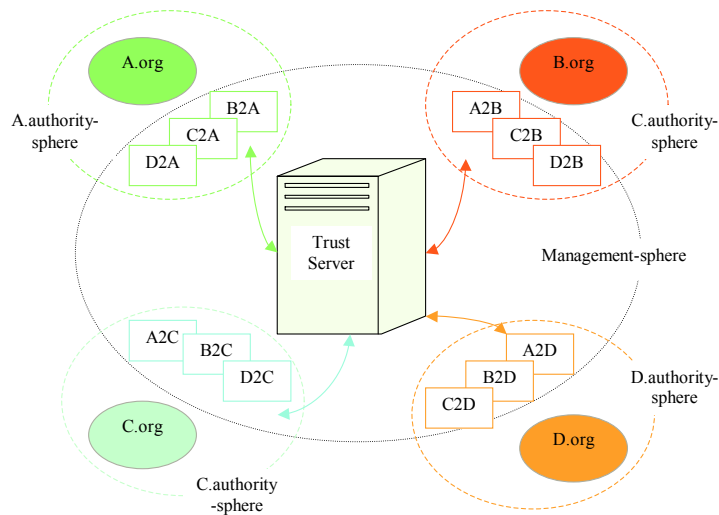
## 7 Application to VO policy administration

In a Virtual Organization (VO), several organizations share some of their subjects, actions and objects to achieve a common purpose. Usually, an initiator organization, which wants to create a VO, will have to issue a query to other organizations it wants to interoperate with. The VO will be created if all the organizations that receive this query agree to be a member of this VO. Each of these organizations will require that the access to its resources must be compliant with some security policy. We claim that these interoperability security policies defined by the different organizations actually correspond to VPOs.

Thus, in our O2O approach, the security policy of the VO is the union of all these VPOs. The problem is then to define how to manage the security policy of the VO. There are three main approaches:

- Decentralized VPO management: This closely corresponds to the approach sketches in figure 1. After defining its VPOs to control interoperability with other organizations

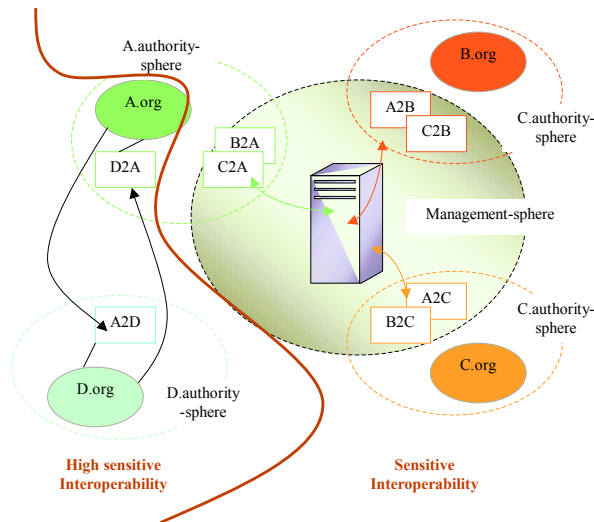
in the VO, each organization will manage these VPOs that are inside its sphere of authority. Thus, when a subject of a given organization A wants to have an access to another organization B, this subject will issue a query. Organization B will apply the VPO A2B to check whether this query is authorized. This will generally require exchanging credentials between A and B for negotiating the access. If this negotiation phase succeeds, then the access will be granted.



**Fig. 2.** Centralized VPO Management

- Centralized VPO management: In this case, a VPO is both in the authority sphere of a given organization which is in charge of defining its interoperability policies and in the *management sphere* of a server (see figure 2) which is in charge of managing all the interoperability policies of those organizations that trust this server. So, managing the VPOs is delegated to a unique trust server, which may be viewed as an extension of a CAS server (Globus toolkit) [16] or an advanced PEP, say PMP for Policy Management Point. Once a VO is created, each organization involved in this VO will have to send its VPOs to this server. When a subject  $s_A$  from a given organization A wants to have an access to another organization B, this subject must send its query to the server. The server will first authenticate this subject to get the proof that this subject is member of one of those organizations involved in the VO. Then the server will apply the VPO A2B and negotiate the access on behalf of organization B. If this negotiation succeeds, the server will sign the query so that the subject can then present this query to organization B for evaluation.
- Hybrid VPO management: The sensitivity of interoperation may vary. In the case of organizations (governmental or military for instance) that deal with high sensitive information, assigning the task of managing the interoperability policies to a server

may not meet the high confidentiality requirements of such organizations. When some Virtual Organization is created, these organizations may not entrust the server used in the Centralized VPO Management approach and/or may not accept to send its interoperability policy to this server because this may disclose some sensitive information and/or may not agree to interoperate with some organizations involved in this VO. In both case, Hybrid VPO management may be used (see figure 3). In this figure, three organizations A.org, B.org and C.org agree to interoperate through Centralized VPO management, whereas the fourth organization D.org only accepts to interoperate with organization A.org using Decentralized VPO management.



**Fig. 3.** Hybrid VPO Management

In every approach, the interoperability policies are specified using the OrBAC model. The authority or the management sphere checks for each query(s,a,o) if a concrete permission for subject s to do action a on the object o can be derived from the specified VPO policies (see [8] for further details on the concrete permission derivation process).

The main advantage of the centralized management approach over the decentralized one is that, since the trust server has a global view of all the VPOs, it can manage possible conflicts between these VPOs. For instance, let us come back to the example of section 5 and consider that a bookshop a grants a special discount on scientific books to students of university b who have a credit card and are more than 18 years old. If the policy of the university b denies the access to the fact that a student is more than 18 years old, then the negotiation between the bookshop a and the university b will always fail. By analyzing the interoperability policies a2b and b2a, the trust server can detect these conflicts and warn the organizations about the impossibility to interoperate in this case.

## 8 Conclusion

Several works investigate interoperability between entities that have not a priori compatible access control policies. The delicate problem that faces both industrialists and researchers is to identify and enforce minimal requirements so that these interactions continue to be compliant with each security policy of those entities involved in this interoperation. In this paper, we claim that most of these works do not establish a clear separation between (1) the definition of the security policy to be applied in this context of interoperation, (2) how to express it, (3) how to administer it and (4) how to manage it. Our O2O approach gives a response to each of these interrogations. We introduce the concept of VPO to designate the sub-organization in charge of the interoperability access control. This interoperability access control policy is constrained by the access control policy of the parent organization of this VPO but perceptibly differs from it. In the O2O approach, VPO policies are expressed using the OrBAC model. Its built-in confinement principle ensure a secure interoperation and its structure around organizations, roles, activities, views and contexts entities makes it easier to specify dynamic fine grained access control. Moreover, the RSSO principle allows a subject to keep his or her role when accessing to another organization.

In the O2O approach, interoperability policies are always defined and administered by the VPO parent organization. In this way, the VPO controls all the external accesses to the resources of the parent organization that is involved in an interoperation. The VO policy can be actually viewed as the union of the VPO policies.

The management of the VPO policies in the O2O approach can be done by the VPO itself (decentralized management) or it can be delegated to a trust server (centralized management) or a combination of the two (hybrid management). In the case of an interoperation involving more than two organizations, the centralized approach is clearly advised as it is able to efficiently manage conflicts and control negotiations. However, the hybrid approach is sometimes necessary when some organizations do not entrust the server used in the centralized approach.

Due to space limitation, we have not tackle, in this paper, the problem of negotiation and the exchange of credentials. Of course, the negotiation protocols like those used by TrustBuilder or Trust-X have to be adapted to handle OrBAC policy expressions.

The collaboration of several organizations in a VO may lead to creation of new objects. Clearly, these new objects do not belong to any members of the VO. Managing accesses to new resources created in a VO represents further work to be done.

**Acknowledgements:** This work was supported by funding from the French RNRT project Politess and by a grant from the GET (Groupe des Ecoles des Télécommunications).

## References

1. A. Abou El Kalam, R. E. Baida, P. Balbiani, S. Benferhat, F. Cuppens, Y. Deswarte, A. Miège, C. Saurel, and G. Trouessin. Organization Based Access Control. In *Policy'03*, Como, Italie, June 2003.

2. M. A. Al-Kahtani and R. Sandhu. A Model for Attribute-Based User-Role Assignment. In *18th Annual Computer Security Applications Conference (ACSAC '02)*, Las Vegas, Nevada, December 2002.
3. E. Bertino, E. Ferrari, and A. Squicciarini. X-TNL: An XML Based Language for Trust Negotiations. In *4th IEEE International Workshop on Policies for Distributed Systems and Networks*, pages 81–84, 2003.
4. S. Cantor, J. Hodges, J. Kemp, and P. Thompson. *Liberty ID-FF Architecture Overview*. Thomas Wason ed., <https://www.projectliberty.org/resources/specifications.php#box1>, 2005. Version 1.2.
5. F. Cuppens, N. Cuppens-Bouahia, T. Sans, and A. Miège. A formal approach to specify and deploy a network security policy. In *Second Workshop FAST*, Toulouse, France, 26-27 August, 2004.
6. F. Cuppens and A. Miège. Administration Model for Or-BAC. *Computer Systems Science and Engineering (CSSE'04)*, 19(3), May, 2004.
7. T. Davies. Spheres of Control. *IBM Systems Journal*, 17:179–198, 1978.
8. O. et al. *The OrBAC Model Web Site*. <http://www.orbac.org>, 2006.
9. I. Foster, C. Kesselman, G. Tsudik, and S. Tuecke. A Security Architecture for Computational Grids. In *5th Conference on Computer and Communications Security*, pages 83–91, San Francisco, CA, 1998.
10. A. Hersberg, Y. Mihaeli, D. Naor, and Y. Ravid. Access Control System Meets Public Infrastructure, Or: Assigning Roles to Strangers. In *IEEE Symposium on Security and Privacy*, Oakland, CA, May 2000.
11. J. Li, N. Li, and W. H. Winsborough. Automated Trust Negotiation Using Cryptographic Credentials. In *12th ACM Conference on Computer and Communications Security (CCS'05)*, November 7-11 2005.
12. M. C. Mont, R. Thyne, K. Chan, and P. Bramhall. Extending HP Identity Management Solutions to Enforce Privacy Policies and Obligations for Regulatory Compliance by Enterprises. In *12th HP OpenView University Association Workshop*, July 10-13 2005.
13. A. Mowshowitz. Virtual organization. *Communications of the ACM*, 40(9):30–37, 1977.
14. R. Oppliger. Microsoft .net passport: A security analysis. *Computer*, 36(7):29–35, 2003.
15. A. Pashalidis and C. J. Mitchell. A Taxonomy of Single Sign-On Systems. In *Lecture Notes in Computer Science*, volume 2727, pages 249 – 264, January 2003.
16. L. Pearlman, C. Kesselman, V. Weich, I. Foster, and S. Tuecke. The Community Authorization Service: Status and Future. In *CHEP03*, La Jolia, CA, March 2003.
17. C. Philips, T. C. Ting, and S. Demurjian. Information Sharing and Security in Dynamic Coalitions. In *SACMAT*, Monterey, CA, June 2002.
18. M. Rittenbruch, H. Kahler, and A. Cremers. Supporting Cooperation in a Virtual Organization. In *ICIS*, 1998.
19. R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-Based Access Control Models. *Computer*, 29(2):38–47, 1996.
20. J. D. Ullman. *Principles of Database and Knowledge-Base Systems, Volume II*. Computer Science Press, 1989.
21. J. Warner, V. Atluri, and R. Mukkamala. A Credential-Based Approach for Facilitating Automatic Resource Sharing among Ad-Hoc Dynamic Coalitions. In *19th Annual IFIP WG 11.3 Working Conference on Data and Applications Security*, Storrs, CT, August 2005.
22. T. Yu, M. Winslett, and K. Seamons. Supporting Structured Credentials and Sensitive Policies through Interoperable Strategies for Automated Trust Negotiations. *ACM Transactions and Information System Security*, 6(1), February 2003.