$DL_{\delta\epsilon}$-OrBAC: Context based Access Control

Narhimene Boustia[1] and Aicha Mokhtari[2]

[1]Saad Dahlab University, Algeria
[2]USTHB University, Algeria

Abstract. The final objective of an access control model is to provide a framework to decide if an action performed by subjects on objects is permitted or not. It is not convenient to directly specify an access control policy using concepts of subjects, objects and actions. In *OrBAC* (Organization Based Access Control), we can not only express static authorizations but also dynamic authorizations, depending on context. Formally, OrBAC is described in first order logic, where the context is one of the argument of predicate. We propose a new formalism based on description logic with *defaults* and *exceptions* [1] to describe and reason on OrBAC model. This paper is an enrichment of a previous work [10] with the introducing of an exception operator (ϵ). This formalism covers not only concepts of information systems like users, objects, subjects and roles but also the context by the add of two operators of default (δ) and exception (ϵ). Notice that time complexity is still polynomial [2].

1 Introduction

Security policy models as DAC [3], MAC [4, 5] and RBAC [6] (Role Based Access Cotrol) provide concepts that are useful in current IS. However, such security policies must also be adapted to deal with new requirements; rule are depending on context.

OrBAC (Organization Based Access Control) is useful to deal with some of these new requirements [7]. OrBAC is a model that allows to express a security policy at organizational level, i.e., indepent from the implementation made for this policy. The access control policy does not directly apply to subject, object and action. It defines static authorizations that apply within organization to control the activities performed by roles on views. However, the OrBAC model also allows specification of more complex dynamic authorizations applying in a given context. The formalism used is the first order logic where the context is an additive argument.

The purpose of this paper is to present $\mathrm{DL}_{\delta\epsilon}$-OrBAC which is a new formalization of OrBAC model based on descrition logic with defaults and exceptions ($\mathcal{AL}_{\delta\epsilon}$) [1]. Description Logic (DLs) [8] are a family of knowledge used to describe and classify concepts and their instances. ($\mathcal{AL}_{\delta\epsilon}$) extends the well known description logic ($\mathcal{AL}$) [8] with the two unary connectives (δ, ϵ) to express respectively default and exception.

The idea is to consider that when we are in a usual context, we obtain a default authorization and we express this fact with the connective (δ), but if the context change, the authorization is excepted, and we express this exception with the connective (ϵ). Notice, that even if we have more thant exception, we can represent this fact with the

Boustia N. and Mokhtari A. (2009).
$\mathrm{DL}\delta\epsilon$-OrBAC: Context based Access Control.
In *Proceedings of the 7th International Workshop on Security in Information Systems*, pages 111-118
DOI: 10.5220/0002175801110118

composition of operators of exception (ϵ), and in spite of this, time complexity still polynomial. This paper is an enrichment of a previous work [10] with the introducing of an exception operator (ϵ).

The rest of the paper is structured as follow. Section 2 presents OrBAC model, section 3 introduces description logic with defaults and exceptions. Section 4 defines $\mathrm{DL}_{\delta\epsilon}$-OrBAC, shows how we express security and how we can infer access control rules in differents contexts. We conclude in section 5 by the prospects of evolution of $\mathrm{DL}_{\delta\epsilon}$-OrBAC.

2 OrBAC Model

The central entity in OrBAC model is *Organization*. An Organization can be seen as an organized group of subjects, each playing a specific role. In the medical domain, "Pière and Marie Curie Center", "Service of Pediatrics",etc are organizations. Subject, Action and Object are respectively abstracted into Role, Activity and View [7].

A *Role* is a set of *Subjects* to which the same security rule apply, for example, the subject "John" plays the role of "Doctor" in the organization "Service of Pediatrics". A *View* corresponds to a set of *Objects* that satisfy a common property, for example, in the medical domain, the view "Medical record" corresponds to the object "Medical record of patient". An *Activity* regroups *Actions* that partake of the same principle. In OrBAC model, Actions will mainly contain computer actions such as "read", "write",etc, when Activities contain "consulting", "writing",etc. *Privileges* only apply in specific *contexts*. Contexts can be used to specify the concrete circumstances where organizations grant roles permission to perform activities on views.

It considers that all actions which are not permitted are prohibited, so it suffice to defines only permission relation.

OrBAC is defined using eight basic sets of entities: **OR** (set of organizations), **S** (set of subjects), **AC** (set of actions), **O** (set of objects), **R** (set of roles), **AV** (set of activities), **V** (set of views) and **C** (set of contexts).

In the next section, we will introduce our formalism used to describe $\mathrm{DL}_{\delta\epsilon}$-OrBAC which is based on description logic with defaults and exceptions.

3 Description Logic with Defaults and Exceptions

Description logic is actually largelly used to represent concept hierarchies, it employs two kinds of formalisms for the knowledge representation: the terminological formalism (TBox) used to describe conceptual knowledge, the assertional formalism (ABox) used to allow facts to be stated [8].

In what follows, we present ($\mathcal{AL}_{\delta\epsilon}$), an extension of $\mathcal{AL}$ language with the two operators of defaults (δ) and exceptions (ϵ).

3.1 $\mathcal{AL}_{\delta\epsilon}$ Language

The description language with defaults and exception $\mathcal{AL}_{\delta\epsilon}$ is inductively defined from a set **R** of primitive roles and a set **P** of primitive concepts [9], augmented by the constant $\top$ (Top), with the abstract syntax rule:

$$
\begin{array}{lll}
C, D \rightarrow \top & & \text{the most general concept} \\
& \mid P & \text{primitive concept} \\
& \mid C \sqcap D & \text{concept conjunction} \\
& \mid \neg P & \text{negation of primitive concept} \\
& \mid \forall\exists r : C & \text{C is a value restriction on all roles R}(> 0) \\
& \mid \delta C & \text{default concept} \\
& \mid C^{\epsilon} & \text{exception to the concept}
\end{array}
$$

δ and ϵ are two unary connectives, $\sqcap$ is a binary conjunction connective and $\forall\exists$ enables universal quantification on role values.

In the next section, we will give the formalization of OrBAC model based on description logic with defaults and exceptions. We will show how these two connectives (δ, ϵ) can be efficient to formalize access control.

4 DL$_{\delta\epsilon}$-OrBAC

We now conceptualize the OrBAC model and construct a DL knowledge base capturing the characteristics of OrBAC, including the context with the use of (δ) and (ϵ).

4.1 The TBox

Given an OrBAC model, we define a DL knowledge base $\mathcal{K}$, the alphabets of $\mathcal{K}$ includes the following atomic concepts: Organization, Subject, Object, Role, View, Action, Context and Activity. The TBox includes the following axioms, each axiom is illustrated with examples.

– **Role Attribution Axiom:** defines the relationship between subject and role.
  $Subject \sqsubseteq \top$; $Role \sqsubseteq \top$;
  $Organization \sqsubseteq \top$
  $Employ \sqsubseteq EmployS.Subject \sqcap EmployR.Role \sqcap EmployOr.Organization$
  Suppose that in a given hospital **X**, **Jean** is assigned in the role of **Doctor**, and **Tom** is assigned in the role of **Surgeon**. We express all these facts by the following rules.
  $Employ(E1) \sqsubseteq EmployS.Subject(Jean) \sqcap EmployR.Role(Doctor)$
  $\sqcap EmployOr.Organization(X)$
  $Employ(E2) \sqsubseteq EmployS.Subject(Tom) \sqcap EmployR.Role(Surgeon)$
  $\sqcap EmployOr.Organization(X)$
  Where E1, and E2 are instances of Employ.

– **View Definition Axiom:** defines relationship between object and view.
  $Object \sqsubseteq \top$
  $View \sqsubseteq \top$
  $Use \sqsubseteq UseO.Object \sqcap UseV.View \sqcap UseOr.Organization$
  Suppose that in a given hospital **X**, **Med-rec1** and **ordinnance1** are instances of concept Object, and **Med-rec** and **ordinnance** are instances of concept View. We express all these facts by the following rules.
  $Use(U1) \sqsubseteq UseO.Object(Med-rec1) \sqcap UseV.view(Med-rec)$
  $\sqcap UseOr.Organization(X)$

$Use(U2) \sqsubseteq UseO.Object(Ordinnance1) \sqcap UseV.view(Ordinnance)$
$\sqcap UseOr.Organization(X)$
Where U1 and U2 are instances of Use.

– **Activity Definition Axiom:** defines relation between action and activity.
$Action \sqsubseteq \top$
$Activity \sqsubseteq \top$
$Consider \sqsubseteq ConsiderAc.Action \sqcap ConsiderAv.Activity$
$\sqcap ConsiderOr.Organization$
Suppose that in a given hospital **X**, action **write** is considered as a modification activity and action **read** as a consultation activity. We express all these facts by the following rules.
$Consider(C1) \sqsubseteq ConsiderAv.Activity(Modify) \sqcap ConsiderAc.Action(write)$
$\sqcap ConsiderOr.Organization(X)$
$Consider(C2) \sqsubseteq ConsiderAv.Activity(Consult) \sqcap ConsiderAc.Action(read)$
$\sqcap ConsiderOr.Organization(X)$
Where C1 and C2 are instances of Consider.

– **Context Definition Axiom:**
$Context \sqsubseteq \top$
$Define \sqsubseteq DefineAc.Action \sqcap DefineS.Subject \sqcap DefineO.Object$
$\sqcap DefineC.Context \sqcap DefineOr.Organization$
We need first to define the **Normal** context.
$Define(D1) \sqsubseteq DefineAc.Action(Write) \sqcap DefineS.Subject(Jean)$
$\sqcap DefineO.Object(Ordinnance) \sqcap DefineC.Context(Normal)$
$\sqcap DefineOr.Organization(X)$
Where D1 is an instance of Define.

– **Permission Attribution Axiom:** defines the relation between role, activity, view and context in an organization.
Prohibition = non Permission
$Permission \sqsubseteq PermisionAv.Activity \sqcap PermissionR.Role$
$\sqcap PermissionV.View \sqcap PermissionC.Context \sqcap PermissionOr.Organization$
Every user who play the role of **Doctor** is permitted to **modify** an **ordinnance** when the context **normal** is true.
$Permission(P1) \sqsubseteq PermisionAv.Activity(Modify)$
$\sqcap PermissionR.Role(Doctor) \sqcap PermissionV.View(Ordinnance)$
$\sqcap PermissionC.Context(Normal) \sqcap PermissionOr.Organization(X)$
Where P1 is an instance of Permission.

– **Hierarchy Definition Axiom:** defines the hierarchy between roles.
$Sub{-}role \sqsubseteq Sub{-}role1.Role \sqcap Sub{-}role2.Role \sqcap Sub{-}roleOr.Organization$
Role1 is a sub role of Role2 in organization Or
Suppose that in a given hospital **X**, a **Surgeon** play also the role of **Doctor**, then a surgeon inherits the set of authorizations of the role doctor. The next rule express this fact.
$Permission(X, Doctor, Av, V, C) \sqsubseteq Permission(X, Surgeon, Av, V, C)$

And because we defined the concept of Sub-role, so the inheritance is expressed as follow:

$Permission(X, Surgeon, Av, V, C) \sqsubseteq Permission(X, Doctor, Av, V, C) \sqcap Sub - role(X, Surgeon, Doctor)$

– **Concrete Permission Axiom:**
  $Is - permitted \sqsubseteq Is - permittedAc.Action \sqcap Is - permittedS.Subject \sqcap Is - pemittedO.Object$
  Jean is permitted to write **Diagnosis**
  $Is - permitted(I1) \sqsubseteq Is - permittedAc.Action(Write) \sqcap Is - permittedS.Subject(Jean) \sqcap Is - pemittedO.Object(Diagnosis)$

Definition of Rules of Security:

$Employ \sqcap Use \sqcap Consider \sqcap \delta Permission \sqcap \delta Define \sqsubseteq \delta Is - permitted$
$Employ \sqcap Use \sqcap Consider \sqcap Permission^{\epsilon} \sqcap Define^{\epsilon} \sqsubseteq Is - permitted^{\epsilon}$

4.2 The ABox

The ABox of $\mathcal{K}$ includes eight catalogs of axioms: Organization assertions axiom, Subject assertions axiom, Object assertions axiom, View assertions axiom, Role assertions axiom, Action assertions axiom, Activity assertions axiom and Context assertions axiom.

In the next section, we show how a security policy can be modelized and how we can infer authorizations.

4.3 Inference and Subsumtion

– **Permission Hierarchy**
  We know that a Surgeon is a sub-role of Doctor, so we can write:
  $Sub - role(S1) \sqsubseteq Sub - role1.Role(Surgeon) \sqcap Sub - role2.Role(Doctor) \sqcap Sub - roleOr.Organization(X)$
  When we want to know if a Surgeon is permitted to write an ordinnance, we use the following rule:
  $Permission(X, Surgeon, Modify, Ordinnance, Normal) \sqsubseteq Permission(X, Doctor, Modify, Ordinnance, Normal) \sqcap Sub - role(S1)$
  and then the answer in this case is **Yes** because we use the inheritance of properties in the normal case.

– **Access Control if Normal Context is True**
  Within organization **X**, **Normal** context holds between subject **Jean**, action **Write** and object **Diagnosis1**, we obtain D2, an instance of concept Define.
  $Define(D2) \sqsubseteq DefineAc.Action(Write) \sqcap DefineS.Subject(Jean) \sqcap DefineO.Object(Diagnosis1) \sqcap DefineC.Context(Normal) \sqcap DefineOr.Organization(X)$
  We also know that **Diagnosis1** is an object used in a view **Diagnosis**, an instance U3 of Use is write as:

$Use(U3) \sqsubseteq UseO.Object(Diagnosis1) \sqcap UseV.view(Diagnosis)$
$\sqcap UseOr.Organization(X)$
And finally, we know that in organization **X**, each person who play the role of **Doctor** is permitted to **modify Diagnosis**, when **Normal** context is true, we write an instance P2 as:
$Permission(P2) \sqsubseteq PermisionAv.Activity(Modify)$
$\sqcap PermissionR.Role(Doctor) \sqcap PermissionV.View(Diagnosis)$
$\sqcap PermissionC.Context(Normal) \sqcap PermissionOr.Organization(X)$
Now, the question is: Is Jean permitted to write diagnosis in a normal context?; We have:
-Jean play role of doctor in organization X: Employ(E1);
-and, Diagnosis1 is an object used in the view Diagnosis: Use(U3);
-and, Write is considered as a modification activity: Consider(C1);
-and, by default, within organization X, context Normal holds between subject Jean, action Write and object Diagnosis1: $\delta Define(D2)$;
-and finally, by default, in organization X, each person who plays the role of Doctor is permitted to modify Diagnosis, when Normal context is true: $\delta Permission(P2)$.
Formally, we write:$Employ(E1) \sqcap Use(U3) \sqcap Consider(C1) \sqcap \delta Permission(P2) \sqcap \delta Define(D2)$
Using security rules, we can deduce that the precedent proposition subsume $\delta Is-permitted(I1)$.
And because $Is-permitted(I1) \sqsubseteq \delta Is-permitted(I1)$, we can deduce that Jean is permitted to write diagnosis.

– **Access Control if Context "Contamination-risk" is True.** We have within organization **X**, context **Contamination-risk** holds between subject **Jean**, action **Write** and object **Diagnosis1**, we obtain D3, an instance of concept Define.
$Define(D3) \sqsubseteq DefineAc.Action(Write) \sqcap DefineS.Subject(Jean)$
$\sqcap DefineO.Object(Diagnosis1) \sqcap DefineC.Context(Contamination-risk) \sqcap DefineC.Context(normal) \sqcap DefineOr.Organization(X)$
We know that context **Contamination-risk** is an exception of context **normal**, $DefineC.Context(Contamination-risk) \equiv (DefineC.Context(Normal))^{\epsilon}$
If we substitute DefineC.Context(Contamination-risk) by its value, we obtain:
$\delta Define(D3) \sqsubseteq \delta DefineAc.Action(Write) \sqcap \delta DefineS.Subject(Jean)$
$\sqcap \delta DefineO.Object(Diagnosis1) \sqcap \delta DefineC.Context(Normal)$
$\sqcap (DefineC.Context(Normal))^{\epsilon} \sqcap \delta DefineOr.Organization(X)$
Using the rule $A^{\epsilon} \equiv \delta A \sqcap A^{\epsilon}$, we obtain:
$(DefineC.Context(Normal))^{\epsilon} \equiv \delta DefineC.Context(Normal)$
$\sqcap (DefineC.Context(Normal))^{\epsilon}$
after replacment, we get:
$\delta Define(D3) \sqsubseteq \delta DefineAc.Action(Write) \sqcap \delta DefineS.Subject(Jean)$
$\sqcap \delta DefineO.Object(Diagnosis1) \sqcap (DefineC.Context(Normal))^{\epsilon}$
$\sqcap \delta DefineOr.Organization(X)$
which means that: $Define(D2)^{\epsilon} \sqsubseteq \delta Define(D3)$
In the context **Contamination-risk**, we create a new instance P3 which is defined as follow:

$Permission(P3) \sqsubseteq PermisionAv.Activity(Modify)$
$\sqcap PermissionR.Role(Doctor) \sqcap PermissionV.View(Diagnosis)$
$\sqcap PermissionC.Context(Normal) \sqcap PermissionC.Context(Contamination-risk) \sqcap PermissionOr.Organization(X)$
Context **Contamination-risk** is an exception of context **normal**
$PermissionC.Context(Contamination-risk) \equiv (PermissionC.Context(Normal))^{\epsilon}$
After substitution, we get:
$Permission(P3) \sqsubseteq PermisionAv.Activity(Modify)$
$\sqcap PermissionR.Role(Doctor) \sqcap PermissionV.View(Diagnosis)$
$\sqcap PermissionC.Context(Normal) \sqcap (PermissionC.Context(Normal))^{\epsilon}$
$\sqcap PermissionOr.Organization(X)$
We know that, $A^{\epsilon} \equiv \delta A \sqcap A^{\epsilon}$, we obtain:
$Permission(P3) \sqsubseteq PermisionAv.Activity(Modify)$
$\sqcap PermissionR.Role(Doctor) \sqcap PermissionV.View(Diagnosis)$
$\sqcap (PermissionC.Context(Normal))^{\epsilon} \sqcap PermissionOr.Organization(X)$
So, we can deduce: $Permision(P2)^{\epsilon} \sqsubseteq \delta Permission(P3)$
The question we can ask is: Is Jean permitted to write Diagnosis when context Contamination-risk is true?
We have: -Jean play role of doctor in organization X: Employ(E1);
-and, Diagnosis1 is an object used in the view Diagnosis: Use(U3);
-and, Write is considered as a modification activity: Consider(C1);
-and, by default, within organization X, context Contamination-risk holds between subject Jean, action Write and object Diagnosis1: $\delta Define(D3)$;
-and finally, by default, in organization X, each person who plays the role of Doctor is permitted to modify Diagnosis, when context Contamination-risk is true: $\delta Permission(P3)$.
We obtain: $Employ(E1) \sqcap Use(U3) \sqcap Consider(C1) \sqcap \delta Permission(P3) \sqcap \delta Define(D3)$
$\equiv Employ(E1) \sqcap Use(U3) \sqcap Consider(C1) \sqcap \delta Permission(P2)^{\epsilon}$
$\sqcap \delta Define(D2)^{\epsilon}$
$\equiv Employ(E1) \sqcap Use(U3) \sqcap Consider(C1) \sqcap Permission(P2)^{\epsilon} \sqcap Define(D2)^{\epsilon}$
Using security rules, we can deduce that the precedent proposition subsume $Is-permitted(I1)^{\epsilon}$. And because $Is-permitted(I1) \not\sqsubseteq Is-permitted(I1)^{\epsilon}$, then we can't deduce Is-permitted(I1), and Jean is not permitted to write diagnosis when there is a contamination risk.

5 Conclusions

The aim of this paper is to give a logical formalisation to DL$_{\delta\epsilon}$-OrBAC model using the expressive $\mathcal{AL}_{\delta\epsilon}$ language. We showed how default and exceptional knowledge are well suited to represent and reason about access control, we illustrate this fact with a small example of medical information system.

To implement and reason about this model, we need to choose a reasoner which take into account default and exceptional knowledge. There are many tools for the classical description logic [11], we mention Classic, Fact++, RacerPro, but in our knowledge, there is no tool for description logic with default and exception reasoner.

The next step of our work is to develop such a reasoner. $Neo-Classic_{\delta\epsilon}$ is actually our preoccupation.

References

1. F. Coupey and C. Fouqueré. Extending conceptual definitions with default knowledge. *Computational Intelligence*, Vol 13, Num 2, 1997.
2. F.M. Donini, M. Lenzerini, D. Nardi, B. Hollunder, W. Nutt and M. Spaccamela. The complexity of existential quantification in concept languages. *Artificial Intelligence*, 53:309-327,1992.
3. B. Lampson. Protection. In *5th Princeton Sympoium on Information Sciences and Systems*, March 1971, pp. 437- 443.
4. D.E. Bell and L.J. LaPadula. Secure computer systems: Unified exposition of multics interpretation. Tech. Rep. ESD-Tr-73-306, The MITRE Corporation, March 1976.
5. K.J. Biba. Integrity consideration for secure computer systems. Tech. Rep. MTR-3153, The MITRE Corporation, June, 1975.
6. R. Sandhu, E.J. Coyne, H.L. Feinstein and C.E. Youman. Role based access control models. In *IEEE Comuter*, Vol. 29, no. 2, pp. 38-47, 1996.
7. A. Abou El Kalam, R. El Baida, P. Balbiani, S. Benferhat, F. Cuppens, Y. Deswarte, A. Miège, C. Saurel, and G. Trouessin. Organization Based Access Control. In *4th IEEE International Workshop on Policies for Distributed Systems and Networks (Policy'03)*, Lake Come, Italie, June 2003.
8. F. Baader, D.L. McGuiness, D. Nardi and P.F. Schneider. The Description logic handbook: Theory, Implementation and Applications. Cambridge university press, 2002.
9. B. Nebel. Reasoning and revision in hybrid representation systems. In *Lecture Note in Computer Science*, Springer-Verlag, 1990.
10. N. Boustia and A. Mokhtari. Representation and reasoning on OrBAC. In *The Third International Conference on Availability, Security and Reliability*, Barcelona, Spain,2008.
11. http://www.cs.man.ac.uk/ sattler/reasoners.html