A Possibilistic Logic Encoding of Access Control

Salem BENFERHAT[1] **Rania EL BAIDA**[2] **Frédéric CUPPENS**[2]

[1] Centre de Recherche en Informatique de Lens (CRIL-CNRS)
Université d'Artois, Rue Jean Souvraz,SP 18 62307 LENS Cedex, FRANCE
E-mail:benferhat@cril.univ-artois.fr

[2] Institut de Recherche en Informatique de Toulouse (IRIT)
Université Paul Sabatier, 118 route de Narbonne 31062 TOULOUSE Cedex 4, FRANCE
E-mail:{elbaida, cuppens}@irit.fr

Abstract

This paper proposes a modelling of information security policies in the framework of possibilistic logic. Our modelling is based on the concept of roles associated with users. Access control rules, guaranteeing the properties of confidentiality and integrity, are encoded in terms of stratified knowledges bases. The stratification reflects the hierarchy between roles and is very useful for dealing with conflicts.

Keywords : possibility theory, security policy, access control, stratified knowledge bases.

Introduction

Modelling information security policies is an important task in many domains. For example, in a health sector, it is very important to guarantee the confidentiality, integrity and availability of pieces of information contained in medical files of patients. For instance, in a medical context, an illegal disclosure or alteration of a patient's record can respectively have serious consequences on patient's reputation or in establishing diagnosis. On the other hand, preventing a physician to have an access to a medical record in an urgency context can have disastrous effects on patient's health. Therefore, it is essential to specify how to control user's access to data. There exist several security policies models for the health care sector (see (Wilikens, Feriti, & Masera 2002) for instance). Most of these models are based on the concept of roles associated with users as in RBAC (*Role-Based Access Control*) (Sandhu *et al.* 1996) or TMAC (*Team-Based Access Control*) (Georgiadis *et al.* 2001) systems. Some role-based systems are not entirely satisfactory. For example, in the RBAC system, a user playing a role of "doctor" may have a permission to read patient's records. If this rule ensures the availability property, it however does not guarantee the confidentiality property. Indeed, in this system it may happen that a doctor has access to all patients' records while only access to patient's records that he is taking care of is desirable.
The joint handling of confidentiality, integrity and availability properties raises the problem of potential conflicts. For instance, a conflict may happen when a policy defines an action that a user is permitted to perform and there exist some situations where performing such action is not acceptable. An example of such conflicting rules is: "patient's families are authorised to read patient's record" and "patient's families are not authorised to read patient's record if the patient explicitly refuses such authorisation".
This paper proposes an access control model based on possibilistic logic. It provides a simple tool to deal with conflicts. Possibilistic logic specifies priority degrees between different security rules.
The access control also uses the concept of roles and users and a new concept that we call care entity. Care entities are introduced in order to assign to each user a least set of privileges necessary to achieve a given task. The second part shows the possibilistic encoding of this model.
We first present basic concepts used in our security policy model. Then, we describe a possibilistic approach of the security model. We show that any consistent set of rules can be transformed into a stratified knowledge base. This stratification enables us to solve conflicts between various rules of the model.

Basic Concepts and Logical Formalisation of Security Policies

The security policy will typically specify access control rules to protect data from unauthorized reading (confidentiality requirement) and unauthorized modification (integrity requirement). Another security requirement is to guarantee that data are accessible and usable upon demand by an authorized user (availability requirement). Ensuring the security of an information system comes down to check that these three requirements are satisfied.
This section presents basic concepts of security policies and shows how they can be encoded. Concepts used in this paper are basically the same as the ones in RBAC and TMAC systems.

Notions of Users, Objects, Actions and Privileges

To illustrate our model, we took an example of a clinic composed of a set of individuals (staffs or patients), called users, and of a set of objects (patients records, rooms, etc). We distinguish several categories of users: patients, their families, insurers and the clinic staffs. We distinguish two kinds of objects: human objects or patients and, non-human objects which can be patient's records, hospital rooms, etc. Patient's records can be also composed of several parts (e.g. patient identification, nurse report ...). Access control can concern the whole objects, or subparts of patient's record.

Actions enable the users to perform access to objects. We shall consider actions on the patient's record and its subparts. They can be represented by the following predicates: ReadRecord(x, y) (i.e., x reads the medical record of y), WriteNurseReport(x, y) (i.e., x writes the nurse report of the medical record of y).

A fourth significant concept is the notion of privileges. Privileges are approvals of a particular mode of access to one or more objects of the system. By a positive privilege, we mean an authorisation (or permission) to execute some action, while a negative privilege refers to prohibition to execute some action.
These concepts are very significant, since the security policy objective is to check if a user within a given context has a privilege to execute some actions on a given objects.

Notions of Roles and Roles Hierarchies

Our model is based on the concept of roles associated with users.
A role is a compact way to assign a set of privileges to a group of users sharing same features. A same user can play different roles. Roles can be dynamically assigned to and removed from users according to changes in organizational structure.
Roles assigned to users are determined by a set of rules having the following form:

If conditions C_i's on (users, contexts) are satisfied then a user U can play a role R

For instance, if a user is recorded as a patient at a clinic then he may play a patient role.
An example of context is: if a user already plays a doctor role and if he exercises his function during night then he may play night's doctor role.

Roles are organised in hierarchies which reflect an organisation's lines of authority and responsibility. Roles hierarchies can be encoded by means of rules having the form:

If a user U plays a role R' then U also plays a role R

Roles hierarchies are used for privileges inheritance. Namely, a user U inherits all privileges (positive or negative) from role R, as soon as he plays a role R'.

Example 1 *Figure 1 gives us an example of hierarchy between roles. For instance, a user playing a surgeon role, plays also doctor role and thus he plays also staff role. Hence, a user playing a role of surgeon, will inherits all privileges assigned to a role of doctor.*

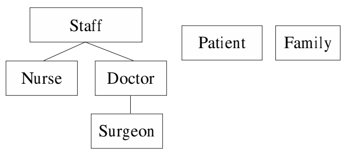

Figure 1: *Roles Hierarchy*

Notions of Care Entities

Care entities are introduced in order to guarantee that only minimal privileges necessary for achieving some tasks are assigned to users. For example, a doctor has access to patient's records only if he is taking care of. For this aim, we need to add, to basic concepts introduced above, the concept of care entity. Care entities are mechanisms restricting the access to objects by users. A care entity is made of: a set of users, a set of objects, the objective of this entity (e.g., a surgical operation, room sterilisation), and temporal information like the beginning and end time of this entity.
The composition of care entities should satisfy some types of constraints. The first type of constraints is related to the care entity objective. It gives necessary conditions for the achievement of the care entity objective. The second type of constraints concerns the separation of roles. These constraints are the same as the one of "mutual exclusion" constraints, between roles, in RBAC system. These constraints say that a same user can not be simultaneously assigned to more than one role (Wilikens, Feriti, & Masera 2002; Sandhu *et al.* 1996).

Example 2 *A care entity whose objective is a surgical operation must be made up of at least one surgeon, one anesthetist, one nurse and one patient.*
To validate this care entity, we add the following constraint of separation of role : the roles surgeon and nurse cannot be activated by the same person in the same care entity.

Norm Base

The norm base describes various types of privileges attribution rules. Basically, these rules have the form:

If Conditions on (users, objects, contexts) are satisfied then a user is permitted / prohibited to execute some actions on (users, objects)

Examples of contexts can be emergency, strike, etc.

Example 3
1. *Non staffs members are not allowed to read patient's records.*
2. *A patient has a permission to read his medical record.*

Logical Formalisation of Security Policies

A formal description of security policies is necessary to check if security properties are satisfied or not.
This section argues that when there is no conflict, classical logic is enough to model security policies. In the following, capital letters refer to predicates symbols, and x, y, z denote

variables. Different roles are then encoded by different predicates, e.g., RDoctor(x) (i.e., x plays a role of doctor), RPatient(x) (i.e., x plays a role of patient) and RFamily(x) (i.e., x plays the role of family). Objects are also encoded by predicates symbols, e.g., MedicalRecord(x), NurseReport(x), etc. The question now is how to encode security policies rules.
Rules of roles attributions and roles hierarchies are represented using classical logical connective operators: $\wedge$ (conjunction), $\vee$ (disjunction), $\neg$ (negation) and $\Rightarrow$ (material implication). For instance, a rule of roles hierarchy specifying that a user playing a doctor role plays also a staff role, is written as follows: $\forall x, RDoctor(x) \rightarrow RStaff(x)$.

We are interested to show how privileges attribution rules can be encoded. For each action A, two predicates are used: "A" (for instance, Read) to say that A is true, and "PA" (e.g., PRead) to say that "permission to execute A" is true.

The confidentiality and the integrity constraints corresponds to check that each time where an action A is true, the permission to execute this action is also true (i.e., PA should be deduced from the knowledge base).
Moreover, we assume that "a prohibition to execute some action A" is interpreted as "a non-permission to execute A". Hence, prohibition rules are translated into non-permission rules.

Example 4

1. *A non staff member is forbidden to read patient's records. In this rule, we use three predicates symbols: $RS(x), RP(y)$ and $PRead(x,y)$ (i.e., x is permitted to read y's medical record). This rule can be encoded as follows:*
   $\forall x, \forall y, \neg RS(x) \wedge RP(y) \rightarrow \neg PRead(x,y)$.
2. *A patient is permitted to read his medical record.*
   This rule can be encoded as follows:
   $\forall y, RP(y) \rightarrow PRead(y,y)$.

Summary of the Policy Model

In this subsection, we summarize the different modules of the security policy model (Figure 2).
The first part is a database which contains the list of staffs

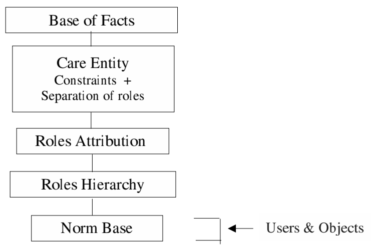

Figure 2: *Security Policy Model*

as well as objects being in the clinic. The second part relates to the validation of the care entity. The third and fourth modules concerns the roles attribution to users as well as the heritage between these roles. The last module is the core of the model; it concerns the privileges attribution rules.
In the following, we will show how this can be encoded in possibilistic logic. But first, we give a brief refresher on possibilistic logic.

Background on Possibilistic Logic

In this section, we recall some basic elements of possibilistic logic, necessary for the reading of this paper (see (Dubois, Lang, & Prade 1994) for more details).
Possibilistic logic is an extension of classical logic. In classical logic, available pieces of information are encoded by means of formulas, having a same level of priority. This makes it difficult to deal with rules having exceptions. Possibilistic logic is a weighted logic where each classical logic formula is associated with a level of priority.

The basic notion in possibilistic logic is called a *possibility distribution*, and denoted by π (Zadeh 1978). A possibility distribution π is a function mapping a set of interpretations Ω into the interval $[0,1]$. $\pi(\omega)$ represents the possibility degree of the interpretation ω with the available beliefs. $\pi(\omega)$ = 1 means that ω is totally possible, $\pi(\omega) > 0$ means that ω is only possible (i.e., ω is not impossible), while $\pi(\omega) = 0$ means that ω is totally impossible.
Given a possibility distribution π, two measures can be defined on the set of propositional formulas ϕ:

- the consistency or possibility degree of ϕ, $\Pi(\phi) = max\{\pi(\omega) : \omega \models \phi\}$, which evaluates to what extent ϕ is consistent with the available beliefs expressed by π;
- the necessity degree of ϕ, $N(\phi) = 1 - \Pi(\neg\phi)$, which evaluates to what extent ϕ is entailed by the available beliefs.

The duality $N(\phi) = 1 - \Pi(\neg\phi)$ extends the existing one in classical logic, where a formula is entailed from a knowledge base if and only if its negation is inconsistent with a knowledge base.
The interval $[0,1]$ can represent a total pre-order between interpretations. In this case, a possibility distribution can be represented in a qualitative form by a partition ($E_1 \cup \ldots \cup E_n$) of Ω where E_1 contains the most plausible interpretations and E_n contains the least plausible interpretations.
At the syntactic level, uncertain pieces of information are represented by a possibilistic knowledge base which is a set of weighted formulas of the form $\Sigma = \{(\phi_i, a_i) : i = 1, n\}$ where ϕ_i is a classical formula and, a_i belongs to $[0,1]$. (ϕ_i, a_i) means that the certainty or priority degree of ϕ_i is at least equal to a_i.
The degrees a_i's can simply express a preference relation between different formulas of the knowledge base. In this case, a possibilistic knowledge base can be put in a stratified form represented by $\Sigma = S_1 \cup \ldots \cup S_n$ such that formulas in S_i have a same certainty level and are less certain than those of S_{i+1}. Thus S_1 contains the least certain formulas and, S_n contains the most ones.

Possibilistic Inference: The notion of inference in possibilistic logic is an extension of the classical logic inference.

To know if a formula ψ is a possibilistic consequence of a knowledge base Σ and an observation ϕ, we first add the formula ϕ in a new layer $S_{n+1} = \{\phi\}$. Let Σ' be the new knowledge base, namely $\Sigma' = \Sigma \cup S_{i+1}$. Namely ϕ is the most certain piece of information. Then we extract a subbase $\delta(\Sigma')$, from Σ' ($\delta(\Sigma') \subseteq \Sigma'$), made of the first important and consistent strata (levels). More formally, $\delta(\Sigma') = S_i \cup \ldots \cup S_{n+1}$, such that $S_i \cup \ldots \cup S_{n+1}$ is consistent but, $S_{i-1} \cup \ldots \cup S_{n+1}$ is inconsistent.
Then ψ is said to be a possibilistic consequence of a knowledge base Σ and ϕ, if $\delta(\Sigma') \vdash \psi$.
The possibilistic inference can be achieved with $log_2(n)$ satisfiability tests, where n is the number of layers in Σ. For sake of simplicity, the rest of examples are given on propositional language.

Example 5 *Let us consider the following stratified base:* $\Sigma = S_1 \cup S_2 \cup S_3$, *where*
$S_1 = \{\neg RS \wedge RP \rightarrow \neg PRead\}$ *(i.e., non-staffs members are not allowed to read patient's medical record),*
$S_2 = \{RP \rightarrow PRead\}$ *(i.e., patients are permitted to read their medical record),*
$S_3 = \{RP \Rightarrow \neg RS\}$ *(i.e., patients are non-staffs members).*
We are interested in checking if a patient is permitted to read his medical record.
By applying the possibilistic inference, we first add RP *in a new strata* $S_4 = \{RP\}$. *Then we extract the first consistent strata* $\delta(\Sigma') = S_4 \cup S_3 \cup S_2$ *(*S_1 *cannot be added since* $\delta(\Sigma') \cup S_1$ *is inconsistent). We can notice that* $\delta(\Sigma') \vdash PRead$

Lexicographical Inference: The possibilistic way of dealing with inconsistency is not entirely satisfactory, since it suffers from an important drawback, named "drowning problem" in (Benferhat *et al.* 1993) as it is illustrated by the following example.

Example 6 *Let us consider previous example, and add the following rules:*
in S_1, $RD \wedge RP \rightarrow PWrite$ *(i.e., doctors are permitted to write patient's record);*
and in S_3*:* $RD \Rightarrow RS$ *(i.e., doctors are staff members);*
and RD *in* S_4.
The possibilistic inference selects the subbase: $\delta(\Sigma') = S_4 \cup S_3 \cup S_2$ *from which* $PWrite$ *cannot be deduced despite the fact that it is not involved in the inconsistency of the knowledge base.*

To take into account the less certain formulas (and which are not responsible of conflicts), a second inference relation, known as lexicographical inference, is defined (Benferhat *et al.* 1993). The idea is to select not only one consistent subbase but several maximally consistent subbases. A consistent subbase $A \subseteq \Sigma$ is said to be lexicographically preferred to a consistent subbase $B \subseteq \Sigma$, denoted by $A >_{Lex} B$, if there doesn't exists a level $n \geq i \geq 1$ such as:

- $|A \cap S_i| > |B \cap S_i|$
- $\forall j > i, j \leq n, |A \cap S_j| = |B \cap S_j|$

We denote by $Lex(\Sigma)$ the set of all preferred consistent lexicographical subbases of Σ' : $Lex(\Sigma) = \{A : A \subseteq \Sigma$ is consistent and $\nexists B \subseteq \Sigma$ consistent, $B >_{Lex} A\}$.
We say that ψ is a lex-consequence of Σ, denoted by $\Sigma \vdash \psi$, if and only of $\forall A \in Lex(\Sigma), A \vdash \psi$. For instance, in example 6, there is exactly one lexicographically preferred subbase:
$A = S_4 \cup S_3 \cup S_2 \cup \{RD \wedge RP \rightarrow PWrite\}$ from which $PWrite$ can be deduced.
The lexicographical inference is more expensive than the possibilistic inference, since it has a complexity of m.SAT where m is the number of formulas in the base. However, several compilations of the lexicographical inference were proposed in (Benferhat *et al.* 2001) and (Coste-Marquis & Marquis 2000) (see also (Darwiche & Marquis 2002)).

Possibilistic logic encoding

Figure 3 summarizes our model. Rules concerning the validation of the care entity can be independently handled in classical logic. Indeed, constraints of care entities composition are hard constraints and thus can be encoded in classical logic.

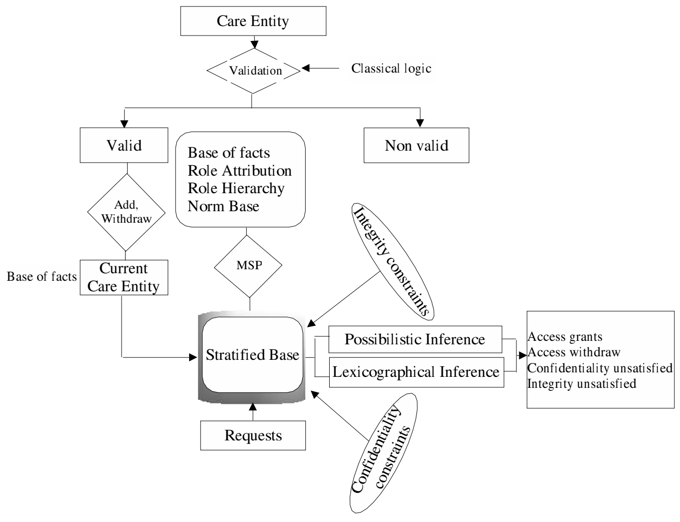

Figure 3: *General Outline*

This section shows how we can automatically extract priorities from the rules that define the security policy. The idea is first to consider that rules without exceptions are always preferred to rules with exceptions. Similary, facts are always preferred to rules with exceptions. We assume that facts together with rules without exceptions are always consistent. Now, the question is how to rank-order rules having exceptions? The idea is to consider that a rule encoding an exceptional situation is preferred to a rule encoding a general situation.

Possibilistic logic offers a simple processing of the rules with exceptions (Benferhat, Dubois, & Prade 1997), which are denoted by "$\alpha \rightarrow \beta$" where " $\rightarrow$ " is a new symbol different from the material implication. An algorithm which transforms a base of rules with exceptions into a stratified knowledge base has been proposed. A rule is considered as "exceptional" if letting its antecedent to be true leads to the inconsistency of the knowledge base.

This algorithm where $W = \emptyset$, is the same than the ones of System Z (Pearl 1990), Possibilistic MSP-inference (Benferhat, Dubois, & Prade 1997), and Lehmann and Magidor's rational closure (Lehmann & Magidor 1992).

Algorithm of Stratification of knowledge bases :
Input : Δ Base of rules with exceptions
W Completely sure rules base
Output: Stratified knowledge base Σ

```
begin
    $m = 1$;
    While$\Delta \neq \emptyset$ do
    begin
        $\Delta^* = \{\neg\alpha_i \vee \beta_i | \alpha_i \rightarrow \beta_i \in \Delta\}$
        $S_m = \{\alpha_i \rightarrow \beta_i | \alpha_i \rightarrow \beta_i \in \Delta \text{ and } \Delta^* \cup W \cup \{\alpha_i\}$
        $\text{is consistent}\}$;
        If $S_m = \emptyset$ then $stop\ (inconsistent\ beliefs)$.
        $\Delta = \Delta - S_m; m = m + 1$;
    end
    Return $\Sigma = S_1 \cup \ldots \cup S_m$.
end
```

Example 7 *Let us take the following rules:* $\Delta = \{\neg S \rightarrow \neg R, P \rightarrow R, S \rightarrow W\}$. *, "generally, non staff members are not allowed to read patients records", "generally, a patient has the permission to read his medical record", "generally, a staff is permitted to write patients records"; and the following completely certain rule:* $W = \{P \Rightarrow \neg S\}$, *"A patient is a non staff member".*
Let us apply the syntactic algorithm to this example. At the first iteration, we obtain $\Delta^* = \{S \vee \neg R, \neg P \vee R, \neg S \vee W\}$*, and* $S_1 = \{\neg S \rightarrow \neg R, S \rightarrow W\}$*. If we repeat this algorithm for the remaining base, we obtain* $\Delta^* = \{\neg P \vee R\}, S_2 = \{P \rightarrow R\}$*. Then* (Δ, W) *is transformed into a stratified base* $S_1 \cup S_2 \cup S_3$*, where* $S_3 = \{P \Rightarrow \neg S\}, S_2 = \{P \rightarrow R\}, S_1 = \{\neg S \rightarrow \neg R, S \rightarrow W\}$.

From the stratified base obtained by this algorithm, we apply either the possibilistic inference or the lexicographical inference for dealing with conflicts.

Conclusion

In this paper, we presented an approach for modelling security policies. Our model is based on the concept of roles to manage the access control making it possible to ensure the objectives and properties of security. We introduced the concept of care entity which guarantees the assignment of least privileges associated with users. We built a stratified beliefs base, from which we can apply either possibilistic inference relation, or lexicographical inference relation.
In this paper, we only consider privileges of the form having the permission or the prohibition to access to the system objects. These privileges as well as the obligation and the recommendation are called simple privileges. There are other forms of privileges, called administrative privileges, which can concern: the activation and deactivation of a simple privilege or role, affectation of a provisional simple role or privilege, creation and deleting roles. These administrative privileges are in general assigned to users who play administrative roles that we did not quote in this paper.

Acknowledgment

This work is supported by the national project RNRT MPSSICSS (Modèles et Politiques de Sécurité des Sytèmes d'Information et de Communication en Santé et en Social). The authors would particularly like to thank P.Balbiani and C.Saurel for their useful discussions, comments and suggestions.

References

Benferhat, S.; Cayrol, C.; Dubois, D.; Lang, J.; and Prade, H. 1993. Inconsistency management and prioritized syntax-based entailment. In *IJCAI'93*, 640–645.

Benferhat, S.; Kaci, S.; Le Berre, D.; and Williams, M. 2001. Weakening Conflicting Information for Iterated Revision and Knowledge Integration. In *IJCAI'01*, 109–118.

Benferhat, S.; Dubois, D.; and Prade, H. 1997. Nonmonotonic reasoning, conditional objects and possibility theory. *Artificial Intelligence Journal*.

Coste-Marquis, S., and Marquis, P. 2000. Compiling Stratified Belief Bases. In *ECAI2000*.

Darwiche, A., and Marquis, P. 2002. Compilation of propositional weighted bases. In *NMR'2002*.

Dubois, D.; Lang, J.; and Prade, H. 1994. Possibilistic logic. In *Handbook of Logic in Artifical Intelligence and Logic Programming*, volume 3, 439–513. Oxford University Press.

Georgiadis, C.; Mavridis, I.; Pangalos, G.; and Thomas, R. 2001. Flexible Team-Based Access Control Using Contexts. In *Sixth ACM Symposium on the Access control Models & Technologies (SACMAT'01)*.

Lehmann, D., and Magidor, M. 1992. What does a conditional knowledge base entail. In *Artificial Intelligence*.

Pearl, J. 1990. System Z: A natural ordering of defaults with tractable applications to default reasoning. In *TARK'90*.

Sandhu, R.; Coyne, E.; Feinstein, H.; and Youman, C. 1996. Role-Based Access Control Models. In *IEEE Computer*, volume 29, 38–47.

Wilikens, M.; Feriti, S.; and Masera, M. 2002. A context-related authorization access control method based on RBAC : a case study from the healthcare domain. In *7th ACM Symposium on Access Control Models and Technologies*.

Zadeh, L. 1978. Fuzzy sets as a basis for a theory of possibility. In *Fuzzy Sets and Systems*, volume 1, 3–28.