

# A Modal Logical Framework for Security Policies

Frédéric Cuppens and Robert Demolombe

ONERA-CERT, 2 Avenue E. Belin, 31055, Toulouse Cedex, France,  
email: {Frederic.Cuppens,Robert.Demolombe}@cert.fr

**Abstract.** It turns out that security becomes more and more important in many information systems. In this paper, we are more specifically interested in confidentiality requirement. In this context, we show how knowledge representation techniques based on formal logic can be used to propose a faithful model of confidentiality. Our approach is to develop a modal logical framework which combines doxastic and deontic logics. This framework enables to specify confidentiality policies –in particular, multilevel security policies– and express various types of security constraints including consistency, completeness and inference control constraints.

**Keywords:** knowledge representation, philosophical foundations, multiagent systems, automated reasoning.

## 1 Introduction

A new interesting application domain for modal logic modelling is multilevel security policies for data base management systems (DBMS). To define these kinds of policies one has to assign a classification level to sentences used to represent data base content and a clearance level to users who can access the data base. Both classification and clearance are taken from a set of security levels which generally has the structure of lattices. Here we focus on the problem of confidentiality of data and the general idea for the definition of a policy is:

*A user can access some data only if the user's clearance level is greater or equal to the classification level of this data.*

In [1] Denning shows that to define a complete model of the security constraints which have to be enforced by a data base management system, it is convenient to decompose the confidentiality problem into two sub-problems:

1. To control information flow inside the DBMS. If this problem is solved we are guaranteed that a user can get information from the DBMS only if he has permission to get it, in the context of the multilevel security policy.
2. To control derived information. If this problem is solved we are guaranteed that a user cannot infer, from the information obtained from the DBMS, information he is not permitted to know.

In this paper we are mainly interested in the second sub-problem, which is usually called the “inference problem”. However, from a theoretical point of view, the separation between the two foregoing problems is not tight; this is because it is difficult to control information flow inside a given DBMS without analyzing what information a user can derive from the information he explicitly gets from the database.

Let us consider a small example to make more concrete what security policies are. We assume that information about some military aircraft is stored in a data base. This information is represented in terms of the following atomic sentences:  $a$ =“the aircraft is equipped with cameras”,  $b$ =“the aircraft is equipped with missiles”,  $c$ =“mission of the aircraft is to perform an air-ground attack versus a given target”. Then, sentence  $a \vee b$ , which is about capacities of the aircraft, is assigned the level 1. Sentences  $a$  and  $\neg a$ , which are about aircraft equipment in a given situation <sup>1</sup>, are assigned the upper level 2. Sentence  $b \rightarrow c$ , which enables to infer what is the aircraft mission, is also assigned the level 2. Finally, sentence  $c$  which represents a critical mission of the aircraft, is assigned the upper level 3. Therefore, the classification levels of sentences are: level 1= $\{a \vee b\}$ , level 2= $\{a, \neg a, b \rightarrow c\}$  and level 3= $\{c\}$ . In this example we can notice that the classification is not complete since, for example, sentence  $b$  has no classification level. A consequence is that, if some user asks the query  $b$  to the DBMS, it is not clear whether the DBMS should answer or not this query. Finally, one might wonder whether the security policy is consistent. In fact we can show that it is not consistent. Indeed, a user cleared at level 2 has permission to access  $a \vee b$  and  $\neg a$ . This allows him to infer  $b$ , and, since he also has permission to access  $b \rightarrow c$ , he can infer  $c$ , which is classified at a level greater than his clearance level.

Most of the difficulties in reasoning about this kind of security policies come from the fact that security policies may be incomplete, in the sense that, for some data, the policy does not explicitly say if users are permitted or not to access these data. Another source of technical difficulties is the fact that the set of data that have a given security level is not necessarily a consistent set. To overcome these difficulties we have designed an appropriate modal logical framework which combines doxastic logic and deontic logic<sup>2</sup>. Notice that the objective of this paper is to find a precise formalization of concepts involved in security policies but it is not our purpose here to design an efficient implementation of the modal logical framework.

In section 2 we present the most important features of the logical framework. Then, in section 3 we show how this framework can be used to formalize the notions of security policy consistency and completeness, and different kinds of

---

<sup>1</sup> Notice that both  $a$  and  $\neg a$  are classified at level 2, but in a given situation, only one of them could be stored in the data base. This means that we consider that a given security policy is defined independently of the data base content. We shall further discuss this point in the remainder of this paper.

<sup>2</sup> It is assumed that the reader is familiar with basic concepts of modal logic. These can be found in [2].

security constraints to prevent inference problems due to deductive reasoning or to abductive reasoning.

## 2 The modal logical framework

### 2.1 Knowledge and Beliefs

We assume that the data base content is represented by a consistent set  $db$  of sentences of a language  $L$  of Propositional Calculus. It is not assumed that sentences in  $db$  are true in the world. This means that  $db$  is considered a set of beliefs.

To analyze confidentiality problems that may occur when accessing a data base, we first need to represent how users interact with the data base. For this purpose, we shall define, for each data base user  $i$ , a modality  $KB_i p$ . Formulae of the form  $KB_i p$ , where  $p$  is a sentence of  $L$ , are to be read: “user  $i$  knows that the data base believes  $p$ ”.

In a structure  $M$ , modality  $KB_i p$  is interpreted by the relation  $R_i$  which is defined on  $W \times W$ , where  $W$  is the set of possible worlds in structure  $M$ . Therefore, the semantics of  $KB_i p$  is defined as follows:

$$M, w \models KB_i p \text{ iff } \forall w' (wR_i w' \Rightarrow M, w' \models p)$$

For convenience the set  $\{w' : wR_i w'\}$  will be denoted by  $R_i(w)$ . The set of worlds where  $p$  is true is denoted by  $\|p\|$ . Using these notations, the above semantics of  $KB_i p$  can be equivalently stated as follows:

$$M, w \models KB_i p \text{ iff } R_i(w) \subseteq \|p\|$$

We shall assume that relation  $R_i$  is serial, that is  $\forall w \in W, R_i(w) \neq \emptyset$ . This means that modality  $KB_i$  obeys the axioms of a KD logic [2].

Notice that the modality  $KB_i$  may be viewed as the combination of an epistemic modality and a doxastic modality which respectively model user’s knowledge and data base beliefs. The epistemic modality may be interpreted by a reflexive accessibility relation (corresponding to a KT logic) and the doxastic modality may be interpreted by another serial accessibility relation (corresponding to a KD logic). In fact, our above serial relation  $R_i$  corresponds to the combination of these two accessibility relations.

### 2.2 Permission and Prohibition to know

We are interested in data bases which contain some sensitive data of the kind presented in the example in section 1. In this context, some users may be not permitted to access the overall data base. What part of the data base an individual is permitted to access is defined by a *confidentiality policy*.

Our formalization of a given confidentiality policy is based on the concept of role [3, 4]. Intuitively, each user can play different roles which define how

he should behave. For each role are defined the permissions, obligations and prohibitions laid upon the role-holder.

In the context of a confidentiality policy, one wants to regulate user's *knowledge*. For this purpose, we shall define, for each role  $r$ , two modalities  $\text{PKB}_r$  and  $\text{FKB}_r$ . Formulae of the form  $\text{PKB}_r p$  (resp.  $\text{FKB}_r p$ ) are to be read: "any agent who plays the role  $r$  is permitted (resp. forbidden) to know that the data base believes  $p$ ".

In a structure  $M$ , modality  $\text{PKB}_r p$  is interpreted by two accessibility relations  $D_r$  and  $R_{xr}$  which are both defined on  $W \times W$ . Intuitively,  $D_r$  is the deontic accessibility relation which characterizes the set of ideal [5, 6] mental states of an agent  $xr$  who plays the role  $r$ . We shall assume that relation  $D_r$  is serial.  $R_{xr}$  is a doxastic accessibility relation which characterizes  $xr$ 's mental state, that is the set of  $xr$ 's beliefs.  $R_{xr}$  is also a serial relation.

Using these two accessibility relations, the semantics of  $\text{PKB}_r p$  is defined as follows:

$$M, w \models \text{PKB}_r p \text{ iff} \\ \exists w' (wD_r w' \text{ and } \forall w'' (w'R_{xr} w'' \Rightarrow M, w'' \models p))$$

or equivalently:

$$M, w \models \text{PKB}_r p \text{ iff } \exists w' (wD_r w' \text{ and } R_{xr}(w') \subseteq \|p\|) \quad (1)$$

Permission is traditionally interpreted as a "possibility" modality. This explains the existential quantifier in definition (1). As usual, prohibition is then defined as negation of permission. Therefore, we have:  $\text{FKB}_r p \stackrel{\text{def}}{=} \neg \text{PKB}_r p$ .

Notice that we can also define, for each role  $r$ , a function  $F_r$  which assigns to each world  $w$  in  $M$  a set of sets of worlds, where each set of world intuitively represents a possible  $xr$ 's mental state. In formal terms,  $F_r$  is a function from  $W$  to  $2^{2^W}$ , which is used to define  $\text{PKB}_r$  as follows:

$$M, w \models \text{PKB}_r p \text{ iff } \exists X (X \in F_r(w) \text{ and } X \subseteq \|p\|) \quad (2)$$

It can easily be shown that definitions (1) and (2) are equivalent, provided the following three conditions hold for all  $w \in W$ :

- $F_r(w) \neq \emptyset$
- $\emptyset \notin F_r(w)$
- $\forall X (X \in F_r(w) \Leftrightarrow \exists w', (wD_r w' \text{ and } R_{xr}(w') = X))$

Notice also that we might define a third modality  $\text{OKB}_r$  for obligation to know as follows:

$$M, w \models \text{OKB}_r p \text{ iff } \forall w' (wD_r w' \Rightarrow R_{xr}(w') \subseteq \|p\|)$$

However, since we are only interested in confidentiality, this modality will not be used in this paper.

### 2.3 Properties of Permission and Prohibition to know

From this semantics, it has been shown in [7] that modalities  $\text{PKB}_r$  and  $\text{FKB}_r$  have the following properties:

**Property 1:**  $\models p \rightarrow q \Rightarrow \models \text{PKB}_r p \rightarrow \text{PKB}_r q$ .

**Property 2:**  $\models p \rightarrow q \Rightarrow \models \text{FKB}_r q \rightarrow \text{FKB}_r p$ .

Properties 1 and 2 show that the set of sentences that an agent playing the role  $r$  is permitted to know is extended by their logical consequences whereas the set of sentences that this agent is forbidden to know is extended by their logical implicants.

We also have the following properties.

**Property 3:**  $\not\models \text{PKB}_r p \wedge \text{PKB}_r q \rightarrow \text{PKB}_r(p \wedge q)$ .

**Property 4:**  $\not\models \text{FKB}_r(p \wedge q) \rightarrow (\text{FKB}_r p \vee \text{FKB}_r q)$ .

The intuitive meaning of property 3 is that it may happen that  $p$  can be derived from a given permitted  $xr$ 's mental state and that  $q$  can be derived from another permitted  $xr$ 's mental state, whereas there is no permitted  $xr$ 's mental state that enables to derive  $p \wedge q$ .

For instance, if, in a model  $M$ , we have:  $F_r(w) = \{\|a\|, \|b\|\}$ , then we have  $M, w \models \text{PKB}_r(a) \wedge \text{PKB}_r(b)$ , whereas we have not  $M, w \models \text{PKB}_r(a \wedge b)$ . This corresponds to a "Chinese wall" type of regulation [8], where the conjunction of two sentences may be strictly more sensitive than each term in the conjunction.

Property 4 is simply the converse of property 3. Its intuitive meaning is that it is possible that an agent playing the role  $r$  is forbidden to know  $p \wedge q$ , whereas he is neither forbidden to know  $p$  nor forbidden to know  $q$ . In fact, it may happen that neither  $p$  nor  $q$  independently enables the derivation of a sensitive sentence, whereas their conjunction enables to derive it.

## 3 Multilevel security policies

### 3.1 Two different approaches

Intuitively, a given multilevel policy assigns a classification level to some sentences of the language used to represent the data base content and a clearance level to data base users. Sentence classifications and user's clearances are both taken in a set of *security levels*. For instance, the *sensitivity levels* Top Secret ( $TS$ ), Secret ( $S$ ), Confidential ( $C$ ), and Public ( $P$ ) may be used as security levels. In this case, the set of sensitivity levels is associated with a total order, viz.  $P < C < S < TS$ . However, it is also possible to define more complex security policies where  $<$  is a partial order (see [9]).

In order to formally represent a given multilevel security policy, we shall define a function which assigns a classification level to some formulas of language  $L$ . To formalize this classification function, we define, for each security level  $l$ , a modality  $[l]$ , and formulae of the form  $[l]p$  are to be read: "formula  $p$  is classified

at level  $l$ ". For instance, the sentence  $[S]\text{Salary}(\text{Smith}, 5000)$  denotes the fact that the classification level of the formula *Smith's salary is 5000* is secret.

We shall assume that the classification function is not necessarily complete: some sentences in  $L$  may not be classified. On the other hand, the set of sentences to which a given classification has been assigned should not be confused with the set of sentences actually stored in the data base. More precisely, we shall assume that the classification function may be defined independently from the multilevel data base content:

1. A sentence may be classified without being stored in the data base. This enables to take into account some general classification rules such as: Smith's salary is secret (and this does not depend on the actual value of Smith's salary stored in the data base). In this case, every formula of the form  $\text{Salary}(\text{Smith}, \text{sal}_i)$ , where  $\text{sal}_i$  is a possible salary value would be classified at secret. But, since we assume that, in a given situation, Smith's salary is unique and, for instance, equal to 5000, only formula  $\text{Salary}(\text{Smith}, 5000)$  will be actually stored in the data base.
2. Conversely, a data base might wrongly behave, for instance by providing some sentence classified at secret to a user whose clearance is confidential. We do not *a priori* exclude that this kind of situation may happen. It will be the purpose of security constraints (see section 4) to decide which situations are satisfactory or not from the point of view of the security policy designer.

For reasons of formalization convenience, instead of modalities  $[l]p$ , we shall actually consider modalities  $[\leq l]$  and  $[\leq \bar{l}]$  (for each security level  $l$ ). These modalities may apply to any finite (not necessarily consistent) set of formulae in  $L$ . Intuitively, formulae of the form  $[\leq l](p_1, \dots, p_n)$  (resp.  $[\leq \bar{l}](p_1, \dots, p_n)$ ) are to be read: " $p_1, \dots, p_n$  is the set of **all** formulae classified at a level lower or equal (resp. classified, but not at a level lower or equal) to  $l$ ".

In a structure  $M$ , these modalities are interpreted by functions  $C_l$  (resp.  $C_{\bar{l}}$ ) from  $W$  into  $2^{2^W}$ . Intuitively if, in world  $w$ , a given formula  $p$  is classified at a level lower or equal to  $l$ , then  $\|p\|$  belongs to  $C_l(w)$ . And if, in world  $w$ , a given formula  $p$  is classified but not at a level lower or equal to  $l$ , then  $\|p\|$  belongs to  $C_{\bar{l}}(w)$ . We shall assume that, if  $l_1 < l_2$ , then, for every  $w \in W$ ,  $C_{l_1}(w) \subseteq C_{l_2}(w)$  and  $C_{\bar{l}_2}(w) \subseteq C_{\bar{l}_1}(w)$ .

Using these functions, the semantics of  $[\leq l](p_1, \dots, p_n)$  and  $[\leq \bar{l}](p_1, \dots, p_n)$  are respectively defined as follows:

$$M, w \models [\leq l](p_1, \dots, p_n) \text{ iff } C_l(w) = \{\|p_1\|, \dots, \|p_n\|\}$$

$$M, w \models [\leq \bar{l}](p_1, \dots, p_n) \text{ iff } C_{\bar{l}}(w) = \{\|p_1\|, \dots, \|p_n\|\}$$

Our next objective is to define permission and prohibition to know from these classification functions. The key point is to notice that we can propose two different attitudes.

The first attitude is to consider that a data base user is permitted to know formulae whose classification level are lower or equal to user's clearance level.

We shall call this first attitude *explicit permission*. It is presented in section 3.2. Prohibition to know is then defined by duality; it corresponds to the notion of *implicit* prohibition.

The second attitude is to consider that a data base user is prohibited to know formulae whose classification level are not lower or equal to the user's clearance level. We shall call this second attitude *explicit prohibition*. It is presented in section 3.3. By duality, we can define the notion of *implicit permission*.

It must be clear that these two attitudes are generally distinct. This is because the classification function is defined in such a way that explicit prohibition does not necessarily coincide with the negation of explicit permission.

### 3.2 Explicit permission

To represent explicit permission in a multilevel security policy, we shall define, for each security level  $l$ , a modality  $\text{PKB}_l^e$ . Formulae of the form  $\text{PKB}_l^e p$  are to be read: "an agent cleared at level  $l$  is explicitly permitted to know that the multilevel data base believes  $p$ ".

As explained in section 2.2, to define the semantics of modality  $\text{PKB}_l^e$  we have simply to define a function  $F_l^1$  from  $W$  into  $2^{2^W}$ . The definition of  $F_l^1$  is derived from function  $C_l$  as follows:

$$F_l^1(w) = \{X : \exists X_1 \exists X_2 \dots \exists X_m \\ \begin{aligned} & ( X_1 \in C_l(w) \text{ and} \\ & X_2 \in C_l(w) \text{ and } \dots \\ & X_m \in C_l(w) \text{ and} \\ & X = X_1 \cap X_2 \cap \dots \cap X_m \text{ and } X \neq \emptyset ) \end{aligned}$$

If we denote by  $f_X$  a formula such that  $\|f_X\| = X$ , then, the constraint on  $X$  intuitively means that  $\vdash f_X \leftrightarrow f_{X_1} \wedge \dots \wedge f_{X_m}$  and  $f_X$  is not a contradiction. That is,  $f_X$  is equivalent to the conjunction of a consistent subset of formulae whose classification levels are less or equal to  $l$ .

We have  $\text{FKB}_l^i p \stackrel{\text{def}}{=} \neg \text{PKB}_l^e p$ .  $\text{FKB}_l^i p$  represents implicit prohibition at level  $l$ .

### 3.3 Explicit prohibition

To represent explicit prohibition in a multilevel security policy, we shall define, for each security level  $l$ , a modality  $\text{FKB}_l^e$ . Formulae of the form  $\text{FKB}_l^e p$  are to be read: "an agent cleared at level  $l$  is explicitly forbidden to know that the multilevel data base believes  $p$ ".

To define the semantics of modality  $\text{FKB}_l^e$  we shall define a second function  $F_l^2$  from  $W$  into  $2^{2^W}$ . Using function  $C_l$ , our function  $F_l^2$  is defined by:

$$F_l^2(w) = \{X : X \subseteq W \text{ and } \forall Y (Y \in C_l(w) \Rightarrow X \not\subseteq Y)\}$$

Condition  $X \not\subseteq Y$  means  $\not\vdash f_X \rightarrow f_Y$ . Then, the constraint on  $X$  means that from  $f_X$  we cannot infer any sentence  $f_Y$  whose classification level is not less or equal to  $l$ .

We have  $\text{PKB}_l^i p \stackrel{\text{def}}{=} \neg \text{FKB}_l^e p$ .  $\text{PKB}_l^i p$  represents implicit permission at level  $l$ .

## 4 Security Constraints Modelling

### 4.1 Consistency of a multilevel security policy

We first present a security constraint to guarantee the confidentiality policy consistency. This constraint is satisfied in a given situation, that is in a given world  $w$  of a model  $M$ , if there is no sentence  $p$  and no security level  $l$  such that a user cleared at level  $l$  is both explicitly permitted to know  $p$  and explicitly forbidden to know  $p$ . The formal representation of this constraint is:

$$M, w \models \neg(\text{PKB}_l^e p \wedge \text{FKB}_l^e p)$$

This may be equivalently stated as follows:

$$M, w \models \text{PKB}_l^e p \rightarrow \text{PKB}_l^i p$$

It is easy to prove that this constraint is enforced if the following condition is satisfied in model  $M$ :

$$\forall X \in F_1^1(w), \exists X' \in F_1^2(w), X' \subseteq X$$

Enforcing the policy consistency constraint is mandatory in every multilevel security policy. This means that the agent who is in charge of defining a multilevel security policy should check whether this policy is consistent.

### 4.2 Completeness of a multilevel security policy

We may also require that the multilevel security policy is complete. Completeness in a given world  $w$  of a model  $M$  means that, for every sentence  $p$  and security level  $l$ , any user cleared at level  $l$  should be explicitly permitted to know  $p$  or explicitly forbidden to know  $p$ . The formal representation of this constraint is:

$$M, w \models \text{PKB}_l^e p \vee \text{FKB}_l^e p$$

This may equivalently be stated as follows:

$$M, w \models \text{PKB}_l^i p \rightarrow \text{PKB}_l^e p$$

Notice that this is the converse of consistency constraint. It is also easy to prove that this constraint is enforced if model  $M$  satisfies the following condition:

$$\forall X \in F_1^2(w), \exists X' \in F_1^1(w), X' \subseteq X$$

If the multilevel security policy is both consistent and complete, then the definitions given in sections (3.2) and (3.3) would be equivalent. This means that explicit permission would coincide with implicit permission. However, we do not consider that enforcing the completeness constraint is mandatory. This means that you may accept that some sentences are neither explicitly permitted nor explicitly prohibited. However, there is a choice which is unavoidable when a user asks for accessing a given sentence. Either you will be *restrictive*, if you actually decide to refuse the access to these sentences, or you will be *permissive*, if you take the opposite decision. In the next section, we shall show how our model enables to represent these two approaches.

### 4.3 Deductive channels

**General constraint** Another general security constraint which has to be satisfied says that, in a given world  $w$  of a model  $M$ , if a user  $i$ , who plays the role  $r$ , knows that the data base believes some sentence  $p$ , then this is permitted for users who play the role  $r$ . This constraint is represented by:

$$M, w \models KB_{i,p} \rightarrow PKB_{r,p} \quad (3)$$

where  $i$  denotes any agent playing the role  $r$ .

It is easy to prove that this constraint is enforced if model  $M$  satisfies the following condition:

$$\exists X \in F_r(w), X \subseteq R_i(w)$$

In the case of a multilevel security policy, we can actually derive two different constraints from this general constraint. The first constraint called *restrictive* is based on explicit permission, whereas the second constraint called *permissive* is based on explicit prohibition. We shall first present the permissive approach and then the restrictive approach.

**Permissive approach** Let us consider a situation where a given user  $i$ , cleared at level  $l$ , gets some data  $p$  from the data base. According to the permissive attitude, this situation is satisfactory if  $p$  is not explicitly prohibited for  $i$  (i.e.,  $\neg FKB_i^e p$ ). In our logic, this first attitude corresponds to the following constraint:

$$M, w \models KB_{i,p} \rightarrow PKB_{i,p}$$

This attitude is the one which is used in every data base which contains some sensitive data and tries to provide users who are not cleared to know the sensitive data with some statistics about them. In this case, this constraint says that a given situation is satisfactory if a given user is provided with data which do not allow him to derive sensitive data.

**Restrictive approach** According to this second attitude, a given situation is satisfactory if a given user  $i$  only knows explicitly permitted data. In our logic, this second attitude corresponds to the following constraint:

$$M, w \models KB_i p \rightarrow PKB_i^c p$$

where  $l$  represents user  $i$ 's clearance level. This second attitude is actually the one which is used in every currently available multilevel data base management system: a given user is only permitted to access some data if this data is really classified at a lower level than user's clearance level. We shall call this constraint restrictive. However, it is interesting to note that this constraint does not necessarily prevent a user from knowing (actually deriving) some data classified at a higher level than this user's clearance, that is, the restrictive attitude does not always imply the permissive attitude. For this purpose, the policy consistency constraint has to be enforced. If it does, it is indeed trivial to prove that, if the policy is consistent, then the restrictive attitude is a stronger attitude than the permissive one.

#### 4.4 Abductive channels

Even if security constraints about deductive channels are enforced, there may be a risk of security policy violation. Indeed, let us consider a situation where agent  $i$  knows that the data base believes  $p$  and  $i$  has permission to know this fact. In this case there is no problem with respect to deductive channels. However, if there exists a sentence  $h$  such that  $p$  and  $h$  implies  $q$  (i.e.  $\vdash h \rightarrow (p \rightarrow q)$ ) and agent  $i$  is not permitted to know that the data base believes  $q$  (i.e.  $\neg PKB_r q$ ), and there is a risk that agent  $i$  knows  $h$  without accessing the data base, then there is a risk of violation for sentence  $q$ .

It is the role of the designer of the security policy to estimate, for a given  $h$ , whether the risk of knowing  $h$  independently of the data base has to be taken into account, but it is the role of some automated reasoning process to determine which sentence  $h$  the designer has to consider.

The predicate  $Abd(p, h, q)$  has been defined in [7] for this purpose.  $Abd(p, h, q)$  holds iff we have  $\vdash h \rightarrow (p \rightarrow q)$  and, to remove trivial cases, the set  $\{p, h\}$  is consistent and we have  $\not\vdash h \rightarrow q$ . Then, the security constraints that prevents abductive channels in a world  $w$  of a model  $M$  is expressed by:

$$M, w \models KB_i p \wedge Abd(p, h, q) \rightarrow PKB_r q \quad (4)$$

where  $i$  denotes any agent playing the role  $r$  and  $h$  is a sentence for which there is a significant risk that agent  $i$  knows  $h$  without accessing the data base.

It is worth noting that, according to the definition of  $Abd(p, h, q)$ , the weakest information agent  $i$  has to know to infer  $q$  from  $p$  is  $p \rightarrow q$ . Then, from a practical point of view, before to provide agent  $i$  with the answer to a query that allows  $i$  to infer  $p$ , it has to be checked whether there exists some sentence  $q$  such that  $\neg PKB_r q$ , and  $p \rightarrow q$  may be known by agent  $i$ .

According on whether we are restrictive or permissive, we can then derive from general constraint (4) two different security constraints to prevent abductive channels in the context of a multilevel security policy.

## 5 Conclusion

We have presented a modal logical framework which can be used to model non trivial security policies. We have seen that security constraints require a fine analysis of two possible attitudes in regard to the regulation; both of them have been formalized in this logic.

The logical framework has been presented from a model theoretic point of view. In [7] has been defined an associated axiomatics which has been proved to be sound and complete if language L contains a finite number of atomic sentences.

Further investigations may go in several directions. The first one is to define efficient strategies for reasoning in this logic in order to design a realistic implementation. This may require to restrict language L to definite Horn clauses.

Another direction is to extend the expressive power of the logic by splitting modalities PKB and FKB into two independent modalities; the first one for permission and prohibition and the second one for knowledge. This extension may be useful to define regulation about access to the regulation itself. For instance, in some applications there is a need for constraints of the form: “agents who play the role  $r_1$  are forbidden to know that agents who play the role  $r_2$  are permitted to know p”; these constraints may be fulfilled using “cover stories” to hide some data (see [10]).

Finally, let us notice that it is sometimes assumed that the security classification of data is determined by the security clearance of users who insert data. For instance in [11] has been presented a model called *causality*, where a user is permitted to know any information deducible from the data this user has inserted in the data base. Such a definition is quite similar to information flow-based definitions of secrecy such as [12, 13]. It would be interesting to show how these information flow control models may be viewed as special cases of the model we present in this paper. For this purpose, we may need to combine deontic modalities with action modalities. This would also enable to consider more general security policies, especially those which combine confidentiality and integrity requirements.

## Acknowledgement

This work was partially supported by the ESPRIT Basic Research Action MED-LAR 2.

## References

1. D. Denning. *Cryptography and Data Security*. Addison-Wesley, 1982.

2. B. F. Chellas. *Modal Logic: An Introduction*. Cambridge University Press, 1988.
3. F. Cuppens. Roles and Deontic Logic. In A. J. I. Jones and M. Sergot, editors, *Second International Workshop on Deontic Logic in Computer Science*, Oslo, Norway, 1994.
4. I. Pörn. *Action Theory and Social Science; Some Formal Models*, volume 120 of *Synthese Library*. D. Reidel, Dordrecht, 1977.
5. J. Carmo and A. J. I. Jones. Deontic database constraints and the characterization of recovery. In A. J. I. Jones and M. Sergot, editors, *Second International Workshop on Deontic Logic in Computer Science*, Oslo, Norway, 1994.
6. A.J.I. Jones and I. Pörn. Ideality, Sub-ideality and Deontic Logic. *Synthese*, 65, 1985.
7. F. Cuppens and R. Demolombe. A Deontic Logic for Reasoning about Confidentiality. In M. Brown and J. Carmo, editors, *Deontic Logic, Agency and Normative Systems*, Workshops in Computing. Springer, 1996.
8. D. Brewer and M. Nash. The Chinese wall security policy. In *IEEE Symposium on Security and Privacy*, Oakland, 1989.
9. C. Landwehr. Formal models for computer security. *ACM Computing Surveys*, 3:247–278, September 1981.
10. F. Cuppens and R. Demolombe. Normative Conflicts in a Confidentiality Policy. In *ECAI'94 Workshop on Artificial Normative Reasoning*, Amsterdam, The Netherlands, 1994.
11. P. Bieber and F. Cuppens. A Logical View of Secure Dependencies. *Journal of Computer Security*, 1(1):99–129, 1992.
12. J. Goguen and J. Meseguer. Unwinding and Inference Control. In *IEEE Symposium on Security and Privacy*, Oakland, 1984.
13. D. Sutherland. A Model of Information. In *Proceedings of the 9th National Computer Security Conference*, 1986.