Towards a formal specification of access control

Mathieu Jaume and Charles Morisset

SPI – LIP6 – University Paris 6,
8 rue du Capitaine Scott, Paris 75015, France,
Mathieu.Jaume@Lip6.fr, Charles.Morisset@Lip6.fr

Abstract. Access control software must be based on a security policy model as software flaws often come from a lack of precision or some incoherences in the policy model. In this paper, we introduce an abstract framework allowing to define access control policies, in a very concise way, offering to refine specifications through several levels and ending by different possible implementations. Such a framework allows to formally reason about security policies and also to compare them, a point which is rarely approached. As an illustration, we give a formal description of the Bell and LaPadula and the Chinese Wall policies and we briefly sketch how to compare these two policies.

Keywords: Formal specifications, Access control policies, Bell and LaPadula, Chinese Wall, Focal.

1 Introduction - Motivations

Security of Information Systems is a major problem for the whole society and is now a well-established field of computer science. The methods used to consider this problem are evolving, as are the ones used in safety areas, where ad-hoc and empirical approaches are being replaced by more formal methods. Presently, for high levels of safety, mathematical models of the requirement/specification phase are mandatory to allow mechanized proofs of the stated properties. In the same way, assurance in system security asks for the use of true formal methods along the process of software development, starting at the specification level [1]. Building a formal library of access control policies needs to formalise in a precise and unambiguous way the access control policies we want to implement (and to prove). Moreover, as it is widely recognised that many access control policies share some common definitions, properties and so proofs, it seems worthwhile to build an abstract generic framework in order to ease and speed implementations by reusing. Indeed, in such a framework, to suit an implementation to a given context would require only the instantiation of some parameters.

The general aim of our work is to develop a formal library of access control policies together with their implementations. For that, we currently use a development environment, namely Focal [13, 6]. Focal provides tools to write specifications, programs, properties and proofs at the same level, proofs being

ultimately checked by Coq [14] and programs compiled to Ocaml [12]. Thanks to these features of Focal, we hope to reach high Evaluation Assurance Levels (EAL-5 6 7) of the Common Criteria, once our model is definitively fixed. Indeed, in previous works [7–9], we have used the Coq proof assistant and the Focal programming environment to provide a formal development of both the Bell and LaPadula model and the "algebra of security" introduced by J.MacLean [11]. Such a development has been used to implement a safe reference monitor in a relational database [3]. Now, we would like to implement the Chinese Wall security policy. However, instead of starting from scratch, we would like to reuse our formal development by following a variant of the lattice-based interpretation of the Chinese Wall policy introduced by Sandhu [15, 16]. Conversely, from a more theoretical point of view, it may be interesting to consider implementations of the Bell and LaPadula policy by reusing a development of the Chinese Wall policy. As we will see, this is not possible and we will define a relation allowing to compare access control policies based on simulations. Note that a simulation based approach has already been considered in [5] to prove equivalence between discretionnary access control models.

Hence, in this paper, we formalise at the level of detail required to obtain a "computer-assisted formalisation" both what is an access control policy and what is the interpretation of a security policy by another one (and what properties such an interpretation must satisfy). Unfortunately, the framework of the "algebra of security" is not expressive enough to achieve this goal and we introduce a specification of access control policies at a deeper level. As the next section will show, such a work also allows to fill the gap between (in)formal papers and (formal) implementations and can be considered as the first step of a formal implementation.

We introduce here an abstract framework allowing to specify access control policies and their implementations. This framework provides several levels of specifications. First, both the entities of the system on which the policy applies and the notion of "secure" states of this system must be specified. Then, requests allowing to get and/or release access (thus allowing to move from a state into another one) must be defined together with a semantics. Of course, there exist several ways more or less restrictive to implement in a "correct" way an access control policy and, last, we have to define a transition function corresponding to an implementation.

An approach based on simulations has already be considered in [5] to prove equivalence between discretionnary access control models.

2 Formalising access control policies

Access control is any mechanism by which a system grants or revokes the rights for active entities, the subjects, to access some passive entities, the objects, or perform some action. Such a mechanism involves two independant concepts: what are the entities (subjects, objects, actions, contexts, roles, ...) and what is the applied security policy (confidentiality, integrity, ...). An extensive literature

on access control exists now (HRU, Bell and LaPadula, Chinese Wall, RBAC, OrBAC, TBAC, TMAC, CBAC, ...) and one can wonder about the usefulness (or the difficulties) of formalising access control policies. However, many papers describing such policies are rather informal and/or present a particular access control mechanism through examples without any formalisation (or generalisation) of the concepts involved in the policy. Of course, such papers are very useful to understand how a particular access control works but they provide little help to implement it in another context. Furthermore, even for papers containing specifications, definitions and properties expressed in a mathematical way, such formalisations are not always done at the level of detail required to obtain a "computer-assisted formalisation". Indeed, even if having a mathematical model drawn by hand is a very serious way to increase confidence, this is not enough. First, nothing is said about how these abstract notions can be implemented. Furthermore, when attempting to check proofs done by hand with a proof assistant, many of them have been discarded. Often, the errors are introduced by points considered as evident details or by forgotten cases. Last, even if proofs done by hand are correct, nothing formally ensures that the implementation meets the "formal" specification done "on the paper".

As a typical example of details that can be crucial, let us consider the classical $\star$-security property introduced to prevent the copy of an object to a lower security level by a malicious subject. If we deal with joint access of groups of subjects over objects, we can "generalize" the property introduced by Bell and LaPadula in [2] as follows:

$$\begin{array}{l}\forall S_1, S_2 \in \wp(\mathcal{S})\backslash\{\emptyset\}\ \forall o_1, o_2 \in \mathcal{O} \\ \quad ((S_1, o_1, A_1 \cup \{\texttt{read}\}) \in m \wedge (S_2, o_2, A_2 \cup \{\texttt{write}\}) \in m \wedge S_1 \cap S_2 \neq \emptyset) \\ \Rightarrow f_o(o_1) \preceq f_o(o_2)\end{array} \tag{1}$$

where $\mathcal{S}$ (resp. $\mathcal{O}$) is a set of subjects (resp. objects), S_1 and S_2 are non-empty sets of subjects, m is a set of access containing elements of the form (S, o, E) expressing that subjects in set S have a joint access to o according to access modes in E and where f_o is a security function that gives the level of security of objects. In [11], this property is stated as follows:

> "*a state is $\star$-secure if for any subjects S_1, S_2 and objects o_1, o_2, if $(S_1, o_1, \{\texttt{read}\}) \in m$ and $(S_2, o_2, \{\texttt{write}\}) \in m$ and the classification of o_1 dominates that of o_2, then $S_1 \cap S_2 = \emptyset$* "

By contraposition of a direct translation of this sentence, we get the following logical formula:

$$\begin{array}{l}\forall S_1, S_2 \in \wp(\mathcal{S})\backslash\{\emptyset\}\ \forall o_1, o_2 \in \mathcal{O} \\ \quad ((S_1, o_1, A_1 \cup \{\texttt{read}\}) \in m \wedge (S_2, o_2, A_2 \cup \{\texttt{write}\}) \in m \wedge S_1 \cap S_2 \neq \emptyset) \\ \Rightarrow \neg(f_o(o_2) \prec f_o(o_1))\end{array} \tag{2}$$

Clearly, since $\preceq$ is only a partial order defining the lattice of security levels, (1) and (2) are not equivalent (they become equivalent when $\preceq$ is a total order,

but most of the time this is not the case). Thus, confusions may arise because these properties actually define different policies. This example points out that different authors give subtly different definitions of the $\star$-security property, and shows that even if these two definitions seem to be similar, by expressing them in a formal way, we obtain two different security policies. Furthermore, note that (2) does not prevent the copy of an object at the level l_1 to a lower security level l_2 (it suffices to write first at a level l_3 which is not comparable to l_1 and l_2). In fact, it is now well-known that formalising specifications, definitions and proofs bring us at a level of detail that is often left to the reader, and by taking into account these details, often considered as minor in informal presentations, definitions and proofs are sometimes getting a little more complicated.

3 Access control policies

3.1 Definitions and Properties

In this paper, we consider access control policies defined over systems viewed as abstract state machines. We give here a very general definition of a security policy $\mathbb{P}[\rho]$ parameterized by some security configuation data ρ (like lattice of security levels, set of conflict-of-interest classes, or anything else), also called *security parameters* in the following.

Definition 1 (Access control policy). *An access control policy $\mathbb{P}[\rho]$, involving security parameters ρ, is defined by a tuple $\mathbb{P}[\rho] = (\mathcal{S}, \mathcal{O}, \mathcal{A}, \Sigma, \Omega)$ where $\mathcal{S}$ is a non-empty set of subjects, $\mathcal{O}$ is a set of objects, $\mathcal{A}$ is a set of access modes, Σ is the set of states of the system on which the policy is defined and Ω is a predicate over states that must be satisfied by states of the system on which the policy applies.*

In this definition, subjects and objects can be viewed as undefined primitive concepts ($\mathcal{S}$ and $\mathcal{O}$ can share some elements). Note that if the policy is correctly implemented, every reachable state must satisfy Ω. In the following, we write $\Sigma_{|\Omega}$ the *set of secure states* $\{\sigma \in \Sigma \mid \Omega(\sigma)\}$. Of course, definitions of Σ and Ω generally depend on ρ, $\mathcal{S}$, $\mathcal{O}$ and $\mathcal{A}$. Furthermore, a state in Σ describes at least the set of current access of the system.

Most of the usual access control policies only consider access control of one single subject over one object. However, it is useful to also deal with "joint access" of a group of subjects over an object as it is done in [11] (for example, to launch a missile, a military system requires that several subjects push a button at the same time). So, we introduce notations allowing to take into account such "joint access" (of course, by constraining group of subjects to be singletons, we get the "usual" notion of access). In this paper, $\mathcal{S}$ stands for a non-empty set of subjects of the system, S stands for an arbitrary non-empty set of subjects ($S \in \wp(\mathcal{S})\backslash\{\emptyset\}$) and s stands for a subject ($s \in \mathcal{S}$). When a set S of subjects jointly gets an access to an object o according to a set of access modes $A \in \wp(\mathcal{A})\backslash\{\emptyset\}$, we write (S, o, A) such *an access* which is an element of the set:

$$\mathbb{M} = (\wp(\mathcal{S})\backslash\{\emptyset\}) \times \mathcal{O} \times (\wp(\mathcal{A})\backslash\{\emptyset\})$$

Note that (S, o, A) means that every subject belonging to S is considered to have an access over the object o. We denote *set of access* by m ($m \in \wp(\mathbb{M})$) and for every m, we require that if (S_1, o_1, A_1) and (S_2, o_2, A_2) belong to m, then either $S_1 \neq S_2$ or $o_1 \neq o_2$. We define an operator $\oplus$ over sets of access allowing to add an access:

$$\oplus : (\wp(\mathbb{M}) \times \mathbb{M}) \rightarrow \wp(\mathbb{M})$$
$$m \oplus (S, o, A) = \begin{cases} (m \backslash (S, o, A')) \cup \{(S, o, A \cup A')\} & \text{if } (S, o, A') \in m \\ m \cup \{(S, o, A)\} & \text{otherwise} \end{cases}$$

Usually, the predicate Ω of an access control policy $\mathbb{P}[\rho]$ is defined by checking that current (and/or past) access satisfy some security properties expressed with security parameters associated with objects and/or subjects. Hence, a state defines both some *security functions*, allowing to give information concerning security parameters of subjects and objects, and information about access of subjects over objects. Note that while security parameters ρ of $\mathbb{P}[\rho]$ are not modified during transitions between states, such security functions (occurring into states) can be modified according to the transformations of states. We introduce a function $\alpha : \Sigma \rightarrow \wp(\mathbb{M})$ such that given a state $\sigma \in \Sigma$, $\alpha(\sigma)$ is the set of access of σ. In the following, we will write $\Sigma_\emptyset$ the set $\{\sigma \in \Sigma \mid \alpha(\sigma) = \emptyset\}$. Last, we introduce notations allowing to consider information over objects and subjects in a state σ. For this, we specify the function $\iota_S(\sigma) : \mathcal{S} \rightarrow \mathcal{I}_S$ (resp. $\iota_O(\sigma) : \mathcal{O} \rightarrow \mathcal{I}_O$) where $\mathcal{I}_S$ (resp. $\mathcal{I}_O$) is a set of information. At this level of specification, we do not specify the structure of elements of $\mathcal{I}_S$ and $\mathcal{I}_O$. Of course, as we will see on examples, ι_S and ι_O can be usually defined from states.

3.2 Examples

In the two following examples, we only focus on mandatory access control.

The Bell and LaPadula policy The Bell and LaPadula model [10, 2] constrains access by considering levels of security associated with subjects and objects. We write $\rho_{BLP} = (\mathcal{L}, \preceq, \sqcup, \sqcap)$ the *lattice of levels of security*. In [10, 2], ρ_{BLP} is defined as the product lattice $T_c \times T_k$ where $T_c = (\mathcal{C}\mathrm{l}, \leq, \sqcup_{cl}, \sqcap_{cl})$ is a lattice of *classifications* and $T_k = (\wp(\mathcal{K}), \subseteq, \cup, \cap)$ is the powerset lattice of a set $\mathcal{K}$ of *needs-to-know*. In this context, a state $\sigma \in \Sigma_{BLP}$ is a tuple $\sigma = (m, f_s, f_o)$ where $\alpha(\sigma) = m$ is the set of current access. The security function $f_s : \mathcal{S} \rightarrow \mathcal{L}$ (resp. $f_o : \mathcal{O} \rightarrow \mathcal{L}$) defines levels of security associated with subjects (resp. objects), f_s and f_o are often called classification vectors. We define the functions ι_S and ι_O as follows:

$$\mathcal{I}_S = \wp(\mathbb{M}) \times \mathcal{L} \qquad \iota_S(\sigma, s) = (\{(S, o, A) \in \alpha(\sigma) \mid s \in S\}, f_s(s))$$
$$\mathcal{I}_O = \wp(\mathbb{M}) \times \mathcal{L} \qquad \iota_O(\sigma, o) = (\{(S, o', A) \in \alpha(\sigma) \mid o' = o\}, f_o(o))$$

The *Bell and LaPadula security policy* is specified by a predicate Ω_{BLP}: $\Omega_{BLP}(\sigma)$ holds iff the property (1) introduced in section 2 and the following property are satisfied.

$$\forall (S, o, A) \in m \quad \texttt{read} \in A \Rightarrow f_s^{\sqcap}(S) \succeq f_o(o)$$

where $f_s^{\sqcap}(S) = \sqcap\{f_s(s) \mid s \in S\}$. We write $\mathbb{P}_{BLP}[\rho_{BLP}] = (\mathcal{S}, \mathcal{O}, \mathcal{A}, \Sigma_{BLP}, \Omega_{BLP})$ such an access control policy.

The Chinese Wall policy The basic idea of the Chinese Wall security policy is that people are only allowed to access information which is not held to conflict with any other information that they already possess. Let $\mathbb{C} = \{c_1, c_2, \cdots, c_n\}$ be a set of *conflict-of-interest classes* and $\mathbb{E}$ be a set of *companies*. Each conflict-of-interest class c_i is associated with a set $f_{\mathbb{E}}(c_i) = \{d_{i1}, d_{i2}, \cdots, d_{ik_i}\}$ of companies that are in "competition" (thus $f_{\mathbb{E}}$ is a function from $\mathbb{C}$ to $\wp(\mathbb{E})$). Conversely, each company d_{ij} belongs to one conflict-of-interest class $c_i = f_{\mathbb{C}}(d_{ij})$ (thus $f_{\mathbb{C}}$ is a function from $\mathbb{E}$ to $\mathbb{C}$). Of course, for all $d \in \mathbb{E}$ and $\forall c \in \mathbb{C}$, we have $d \in f_{\mathbb{E}}(c)$ iff $f_{\mathbb{C}}(d) = c$. *Sanitized* (or public or non-sensitive) information can be treated by considering a particular conflict-of-interest class c_0 which is a special class containing only one company d_0 corresponding to the sanitized company. We are now in position to define security parameters for the Chinese Wall access control policy as $\rho_{CW} = (\mathbb{C}, \mathbb{E}, f_{\mathbb{E}})$. In the following, we will write π_i the projection operator such that $\pi_i(e_1, e_2, \cdots, e_n) = e_i$. Each object can be viewed as a container of information and we introduce the security function:

$$f_o : \mathcal{O} \to \prod_{i=1}^{n} (f_{\mathbb{E}}(c_i) \cup \{\bot\})$$

such that given an object $o \in \mathcal{O}$, $f_o(o) = (e_1, \cdots, e_n)$ means that the object o contains information from companies in the set of companies $\{e_1, e_2, \cdots, e_n\} \backslash \{\bot\}$. Hence, either $\pi_i(f_o(o)) = \bot$, and in this case o does not contain any information from a company of the conflict-of-interest class c_i, or $\pi_i(f_o(o)) = d_{ij} \neq \bot$ is a company in $f_{\mathbb{E}}(c_i)$ and o contains information from this company. Of course, with such notations, an object cannot contain information from two companies of the same conflict-of-interest. For an object o containing information from at least one company which is not the sanitized company d_0, $f_o(o)$ does not give any supplementary information concerning the sanitized information potentially contained by o and for an object o only containing sanitized information, we have $f_o(o) = (\bot, \cdots, \bot)$. A state for the Chinese Wall policy can be described by a set of current access and a security function f_o. Hence, we have:

$$\Sigma_{CW} = \wp(\mathbb{M}) \times \left\{ f_o : \mathcal{O} \to \prod_{i=1}^{n} (f_{\mathbb{E}}(c_i) \cup \{\bot\}) \right\}$$

The functions ι_S and ι_O can be defined as follows:

$$\begin{array}{ll} \mathcal{I}_S = \wp(\mathbb{M}) & \iota_S(\sigma, s) = \{(S, o, A) \in \alpha(\sigma) \mid s \in S\} \\ \mathcal{I}_O = \wp(\mathbb{M}) \times \prod_{i=1}^{n} (f_{\mathbb{E}}(c_i) \cup \{\bot\}) & \iota_O(\sigma, o) = (\{(S, o', A) \in \alpha(\sigma) \mid o' = o\}, f_o(o)) \end{array}$$

While in a strict reading of [4], an object o can be viewed as an atomic information of one single company, as it is done in [15, 16], it could be interesting to consider objects containing information coming from several companies in different conflict-of-interest classes. In this case, the *Chinese Wall security policy*

can be formally specified by the predicate Ω_{CW} defined as follows. Given a state $\sigma = (m, f_o) \in \Sigma_{CW}$, $\Omega_{CW}(\sigma)$ holds iff the two following conditions are stisfied:

$$\forall (S_1, o_1, A_1), (S_2, o_2, A_2) \in m$$
$$S_1 \cap S_2 \neq \emptyset \Rightarrow \forall i \ (1 \leq i \leq n) \ \begin{pmatrix} \pi_i(f_o(o_1)) = \bot \vee \pi_i(f_o(o_2)) = \bot \\ \vee \pi_i(f_o(o_1)) = \pi_i(f_o(o_2)) \end{pmatrix}$$

$$\forall (S_1, o_1, A_1 \cup \{\texttt{write}\}), (S_2, o_2, A_2 \cup \{\texttt{read}\}) \in m$$
$$S_1 \cap S_2 \neq \emptyset \Rightarrow \begin{pmatrix} f_o(o_2) = (\bot, \cdots, \bot) \\ \vee \ \forall i \ (1 \leq i \leq n) \ (\pi_i(f_o(o_2)) = \bot \vee \pi_i(f_o(o_1)) = \pi_i(f_o(o_2))) \end{pmatrix}$$

We write $\mathbb{P}_{CW}[\rho_{CW}] = (\mathcal{S}, \mathcal{O}, \mathcal{A}, \Sigma_{CW}, \Omega_{CW})$ such an access control policy.

3.3 Comparing access control policies (1)

In this subsection, we address the problem of comparing access control policies at this level of specification (next section follows this discussion by considering implementations of access control policies). Indeed, when two different access control policies apply over the same subjects and objects with the same access modes, it can be interesting to compare them. Of course, many points of view can be considered. The most naive way to compare two policies $\mathbb{P}_1[\rho_1] = (\mathcal{S}, \mathcal{O}, \mathcal{A}, \Sigma_1, \Omega_1)$ and $\mathbb{P}_2[\rho_2] = (\mathcal{S}, \mathcal{O}, \mathcal{A}, \Sigma_2, \Omega_2)$ is to say that $\mathbb{P}_1[\rho_1] \leqq \mathbb{P}_2[\rho_2]$ iff:

$$\exists \kappa_\Sigma \subseteq \Sigma_1 \times \Sigma_2 \quad \forall \sigma_1 \in \Sigma_1 \ \Omega_1(\sigma_1) \Rightarrow (\exists \sigma_2 \in \Sigma_2 \ \kappa_\Sigma(\sigma_1, \sigma_2) \wedge \Omega_2(\sigma_2))$$

Intuitively, $\mathbb{P}_1[\rho_1] \leqq \mathbb{P}_2[\rho_2]$ means that $\mathbb{P}_1[\rho_1]$ is more restrictive than $\mathbb{P}_2[\rho_2]$ and $\kappa_\Sigma(\sigma_1, \sigma_2)$ means that σ_1 and σ_2 are "equivalent" or "similar" states. However, such an approach does not take into account the "behavior" of the implementation of the access control policy (its behavioural model) and it suffices to consider $\kappa_\Sigma = \Sigma_1 \times \Sigma_2$ to prove that $\mathbb{P}_1[\rho_1] \leqq \mathbb{P}_2[\rho_2]$ and similarly $\mathbb{P}_2[\rho_2] \leqq \mathbb{P}_1[\rho_1]$. Hence, it may be more adequate to say that $\mathbb{P}_1[\rho_1]$ is more restrictive than $\mathbb{P}_2[\rho_2]$ iff each implementation of $\mathbb{P}_1[\rho_1]$ can be simulated by an implementation of $\mathbb{P}_2[\rho_2]$. To achieve this goal, we have to define what are the implementations of an access control policy, then what are properties such implementations must satisfy and last what is a simulation. However, by following this approach, it may be difficult to prove that $\mathbb{P}_1[\rho_1]$ is more restrictive than $\mathbb{P}_2[\rho_2]$ since there potentially exist many implementations of $\mathbb{P}_1[\rho_1]$. Hence, a possible way to solve this problem is to only consider less restrictive implementations (i.e. maximal) of $\mathbb{P}_1[\rho_1]$. We present such an approach in the next section.

4 Implementations of access control policies

4.1 Requests

The notion of access control policy introduced in the previous section allows to specify what are the entities of the system and what are the "secure" states of

this system. Transformations of states of a system are due to users which submit requests in order to access objects. In the following we will write $\mathcal{R}$ the set of requests.

Example Often, we consider languages of requests allowing to express at least the 4 following kinds of requests:

- the set of subjects S asks for permission to read (resp. to write) an object o, we write $\langle +, S, o, \texttt{read}\rangle$ (resp. such a request $\langle +, S, o, \texttt{write}\rangle$) such a request,
- the set of subjects S asks for permission to stop read access (resp. write access) on an object o, we write $\langle -, S, o, \texttt{read}\rangle$ (resp. $\langle -, S, o, \texttt{write}\rangle$) such a request.

In all the paper, we will consider the following set of requests to study various access control policies.

$$\mathcal{R} = \{\langle +, S, o, \texttt{read}\rangle, \langle +, S, o, \texttt{write}\rangle, \} \tag{3}$$

Hence, we do not treat requests allowing to remove access. Indeed, even if the classical model of the Bell and LaPadula policy takes into account releasing of access, this is not the case for the classical Chinese Wall policy and, as we want to compare these two policies, this can only be done over the same set of requests. Of course, another possibility to make this comparison could consist in considering releasing of access both for the Bell and LaPadula policy and the Chinese Wall policy.

Given a set $\mathcal{R}$ of requests, the "meaning" of requests must be specified. Indeed, comparing implementations of an access control policy makes sense only if they are based on the same semantics of requests. For example, comparing a "classical" implementation of a policy with an implementation that treats `read` as `write` and `write` as `read` does not make much sense. So, we define semantics of requests as a relation as follows.

Definition 2. *Given an access control policy $\mathbb{P}[\rho] = (\mathcal{S}, \mathcal{O}, \mathcal{A}, \Sigma, \Omega)$, the semantics of a set of requests $\mathcal{R}$ is a relation $[\![\mathcal{R}]\!]_\Sigma \subseteq \Sigma \times \mathcal{R} \times \mathcal{D} \times \Sigma$ where $\mathcal{D} = \{\mathsf{yes}, \mathsf{no}\}$, such that: $\forall \sigma \in \Sigma\ \forall R \in \mathcal{R}\ (\sigma, R, \mathsf{no}, \sigma) \in [\![\mathcal{R}]\!]_\Sigma$*

Example For example, we can define the semantics $[\![\mathcal{R}]\!]_{\Sigma_{BLP}}$ for the set $\mathcal{R}$ defined in (3) as follows:

$$\begin{array}{l} (\sigma, \langle +, S, o, \mathtt{x}\rangle, \mathsf{yes}, \sigma') \in [\![\mathcal{R}]\!]_{\Sigma_{BLP}} \Leftrightarrow \left(\begin{array}{c} m' = m \oplus (S, o, \{\mathtt{x}\}) \\ \wedge\, \forall o' \in \mathcal{O}\ f_o(o') = f'_o(o') \end{array}\right) \\ (\sigma, \langle +, S, o, \mathtt{x}\rangle, \mathsf{no}, \sigma') \in [\![\mathcal{R}]\!]_{\Sigma_{BLP}} \Leftrightarrow \sigma = \sigma' \end{array} \tag{4}$$

where $\mathtt{x} \in \{\texttt{read}, \texttt{write}\}$, $\sigma = (m, f_s, f_o)$ and $\sigma = (m', f'_s, f'_o)$. According to this semantics, if the request for an access is granted, then this access has to be added or removed (depending on the request), and no other modification can be done except the change of the levels of security of subjects. Conversely, if the request is denied, then the state does not change. The definition of $[\![\mathcal{R}]\!]_{\Sigma_{CW}}$ is similar with $\sigma = (m, f_o)$ and $\sigma = (m', f'_o)$.

4.2 Implementations of access control policies

Given a set $\mathcal{R}$ of requests, implementing the policy $\mathbb{P}[\rho] = (\mathcal{S}, \mathcal{O}, \mathcal{A}, \Sigma, \Omega)$, as an abstract state machine, consists in defining both a set Σ_I of *initial states* and a *transition function* τ which allows to move from a state to another state of the system according to a request in $\mathcal{R}$.

Definition 3 (Implementations of an access control policy). *An implementation of an access control policy $\mathbb{P}[\rho] = (\mathcal{S}, \mathcal{O}, \mathcal{A}, \Sigma, \Omega)$ is a pair (τ, Σ_I) where $\Sigma_I \subseteq \Sigma_\emptyset \cap \Sigma_{|\Omega}$ is a set of initial states and where $\tau : \mathcal{R} \times \Sigma \to \mathcal{D} \times \Sigma$ is a transition function.*

We write $\Gamma_\tau^n(E)$ (resp. $\Gamma_\tau(E)$) the set of states of a system that are reachable by applying n times (resp. by applying a finite number of times) the function τ from an initial state $\sigma \in E$.

Properties We now define the following properties over implementations of a policy $\mathbb{P}[\rho] = (\mathcal{S}, \mathcal{O}, \mathcal{A}, \Sigma, \Omega)$.

- The implementation (τ, Σ_I) is said to be $\mathbb{P}[\rho]$-*correct* iff each state reachable from a initial state is secure:
$$\mathbb{P}[\rho] \vdash (\tau, \Sigma_I) \Leftrightarrow \Gamma_\tau(\Sigma_I) \subseteq \Sigma_{|\Omega}$$
- The implementation (τ, Σ_I) is said to be $\mathbb{P}[\rho]$-*complete* iff each secure state is reachable from a initial state:
$$\mathbb{P}[\rho] \models (\tau, \Sigma_I) \Leftrightarrow \Sigma_{|\Omega} \subseteq \Gamma_\tau(\Sigma_I)$$
- The implementation (τ, Σ_I) is said to be $\mathcal{R}$-*correct* according to $[\![R]\!]_\Sigma$ iff:
$$[\![R]\!]_\Sigma \vdash (\tau, \Sigma_I) \Leftrightarrow \begin{pmatrix} \forall \sigma \in \Sigma_{|\Omega}\ \forall \sigma' \in \Sigma\ \forall a \in \mathcal{D}\ \forall R \in \mathcal{R} \\ \tau(R, \sigma) = (a, \sigma') \Rightarrow (\sigma, R, a, \sigma') \in [\![R]\!]_\Sigma \end{pmatrix}$$

We write $\mathbb{P}[\rho], [\![R]\!]_\Sigma \vdash (\tau, \Sigma_I)$ when (τ, Σ_I) is both $\mathbb{P}[\rho]$- and $\mathcal{R}$-correct.

In order to be able to compare implementations in a meaningful way, we also introduce a property characterizing implementations of an access control policy allowing the same actions for subjects or objects associated with the same information. Such implementations are called *non-discriminatory* implementations. In the rest of the paper we only consider non-discriminatory implementations.

Definition 4. *The implementation (τ, Σ_I) is said to be non-discriminatory iff:*

$$\begin{array}{l} \forall \sigma, \sigma_1, \sigma_2 \in \Sigma\ \forall s_1, s_2 \in \mathcal{S}\ \forall o_1, o_2 \in \mathcal{O}\ \forall R_1, R_2 \in \mathcal{R}\ \forall d_1, d_2 \in \mathcal{D} \\ \quad \left(\begin{array}{l} \quad \iota_S(\sigma, s_1) = \iota_S(\sigma, s_2) \wedge \iota_O(\sigma, o_1) = \iota_O(\sigma, o_2) \\ \wedge\ \tau(R_1, \sigma) = (d_1, \sigma_1) \wedge \tau(R_2, \sigma) = (d_2, \sigma_2) \\ \wedge\ R_2 = R_1\,[s_1 \leftrightarrow s_2; o_1 \leftrightarrow o_2] \end{array} \right) \\ \Rightarrow \left(\begin{array}{l} \quad d_1 = d_2 \\ \wedge\ \iota_S(\sigma_1, s_1) = \iota_S(\sigma_2, s_2) \wedge \iota_O(\sigma_1, o_1) = \iota_O(\sigma_2, o_2) \\ \wedge\ \iota_S(\sigma_1, s_2) = \iota_S(\sigma_2, s_1) \wedge \iota_O(\sigma_1, o_2) = \iota_O(\sigma_2, o_1) \end{array} \right) \end{array}$$

where $R_2 = R_1\,[s_1 \leftrightarrow s_2; o_1 \leftrightarrow o_2]$ means that R_2 is obtained from R_1 by swapping s_1 and s_2 and by swapping o_1 and o_2.

Preorder over implementations Given two implementations (τ_1, Σ_I^1) and (τ_2, Σ_I^2) of the same access control policy, we say that (τ_1, Σ_I^1) is *more restrictive* (in terms of reachable states) than (τ_2, Σ_I^2), and in this case we will write $(\tau_1, \Sigma_I^1) \sqsubseteq_\Gamma (\tau_2, \Sigma_I^2)$, if and only if every reachable state with (τ_1, Σ_I^1) is also reachable with (τ_2, Σ_I^2).

$$(\tau_1, \Sigma_I^1) \sqsubseteq_\Gamma (\tau_2, \Sigma_I^2) \Leftrightarrow \Gamma_{\tau_1}(\Sigma_I^1) \subseteq \Gamma_{\tau_2}(\Sigma_I^2)$$

Clearly, $\sqsubseteq_\Gamma$ defines a preorder relation over implementations of an access control policy.

Equivalent states As we will see in the next subsection, it may be useful to define some properties over implementations given an equivalence relation $\equiv$ over states. We will say that a transition function τ is $\equiv$-preserving iff the relation $\equiv$ is a congruence for τ.

Definition 5. *Given an equivalence relation $\equiv$ over states and an answer $a \in \mathcal{D}$, a transition function τ is $\equiv_a$-preserving iff:*

$$\begin{aligned} &\forall \sigma_1, \sigma_1', \sigma_2 \in \Sigma\ \forall R \in \mathcal{R}\ (\sigma_1 \equiv \sigma_2 \wedge \tau(R, \sigma_1) = (a, \sigma_1')) \\ \Rightarrow &\exists \sigma_2' \in \Sigma\ (\tau(R, \sigma_2) = (a, \sigma_2')) \wedge \sigma_1' \equiv \sigma_2') \end{aligned}$$

Moreover, τ is said to be $\equiv$-preserving iff τ is $\equiv_a$-preserving for all $a \in \mathcal{D}$.

Given an equivalence relation $\equiv$ over states, it is also possible to define a preorder relation $\sqsubseteq_\equiv$ over transition functions. Roughly speaking, $\tau_1 \sqsubseteq_\equiv \tau_2$ means that τ_2 "simulates" τ_1 for positive answers and τ_1 is $\equiv$-preserving for negative answers.

Definition 6. *Given an equivalence relation $\equiv$ over states and two transition functions τ_1 and τ_2, $\tau_1 \sqsubseteq_\equiv \tau_2$ iff τ_1 is $\equiv_{\mathsf{no}}$-preserving and*

$$\begin{aligned} &\forall \sigma_1, \sigma_2, \sigma_1' \in \Sigma\ \forall R \in \mathcal{R}\ (\sigma_1 \equiv \sigma_2 \wedge \tau_1(R, \sigma_1) = (\mathsf{yes}, \sigma_1')) \\ \Rightarrow &\exists \sigma_2' \in \Sigma\ (\tau_2(R, \sigma_2) = (\mathsf{yes}, \sigma_2') \wedge \sigma_1' \equiv \sigma_2') \end{aligned}$$

In the following we will write $(\tau_1, \Sigma_I^1) \sqsubseteq_{\Gamma,\equiv} (\tau_2, \Sigma_I^2)$ when $(\tau_1, \Sigma_I^1) \sqsubseteq_\Gamma (\tau_2, \Sigma_I^2)$ and $\tau_1 \sqsubseteq_\equiv \tau_2$. We prove the following result.

Lemma 1. *Given two transition functions τ_1 and τ_2 such that $\tau_1 \sqsubseteq_\equiv \tau_2$ for an equivalence relation $\equiv$ over Σ, if τ_2 is $\equiv$-preserving, then τ_1 is also $\equiv$-preserving.*

Proof. Let $\sigma_A, \sigma_B, \sigma_C \in \Sigma$ such that $\sigma_A \equiv \sigma_B$, $R \in \mathcal{R}$ and $a \in \mathcal{D}$. Let us prove that if $\tau_1(R, \sigma_A) = (a, \sigma_C)$ then there exists a state σ_D such that $\tau_1(R, \sigma_B) = (a, \sigma_D)$ with $\sigma_C \equiv \sigma_D$. Two cases are possible. If $a = \mathsf{no}$, then since $\sigma_A \equiv \sigma_B$ and $\tau_1 \sqsubseteq_\equiv \tau_2$, we can conclude by definition of $\sqsubseteq_\equiv$. Conversely, if $a = \mathsf{yes}$, then let σ_D be a state such that $\tau_1(R, \sigma_B) = (a', \sigma_D)$ for an answer a'. Since $\tau_1 \sqsubseteq_\equiv \tau_2$, by definition of $\sqsubseteq_\equiv$ we have $a' = \mathsf{yes}$. Indeed, since τ_1 is $\equiv_{\mathsf{no}}$-preserving, $a' = \mathsf{no}$ implies $a = \mathsf{no}$ which is contradictory. Moreover, we clearly have $\sigma_A \equiv \sigma_A$ and then $\tau_2(R, \sigma_A) = (\mathsf{yes}, \sigma_E)$ and $\sigma_C \equiv \sigma_E$ for a state σ_E. Similarly, $\tau_2(R, \sigma_B) = (\mathsf{yes}, \sigma_F)$ and $\sigma_D \equiv \sigma_F$ for a state σ_F. Since, by hypothesis, τ_2 is $\equiv$-preserving and $\sigma_A \equiv \sigma_B$ we have $\sigma_E \equiv \sigma_F$. Now, by transitivity of $\equiv$, we have $\sigma_C \equiv \sigma_D$ which allows to conclude.

4.3 Examples

Implementations of the Bell and LaPadula policy In [10, 2], Bell and LaPadula introduce an implementation of their policy by defining a transition function τ_{BLP} based on a language of requests richer than the one we consider along this paper. Indeed, access modes considered in [10, 2] are not only `read` and `write` like in the language defined in (3) and as we said, releasing of access is also considered. Moreover, the language of requests used by Bell and LaPadula is such that requests are only submitted by one subject (groups of subjects are constrained to be singletons). Then, the authors prove the famous "Basic Security Theorem":

Proposition 1 ([10]). *Let σ, σ' be two states in:*

$$\{\sigma_s \in \Sigma_{BLP} \mid \forall (S, o, A) \in \alpha(\sigma_s)\ \mathsf{Card}(S) = 1\} \subseteq \Sigma_{BLP}$$

If $\Omega_{BLP}(\sigma)$ and $\tau_{BLP}(R, \sigma) = (a, \sigma')$ (where R is a request submitted by a single subject), then $\Omega_{BLP}(\sigma')$.

Note that in our examples, we only consider mandatory properties and we introduce the following restriction of τ_{BLP}.

$$\tau'_{BLP}(R, (m, f_s, f_o))$$

$$= \begin{cases} (\mathsf{yes}, (m \oplus (\{s\}, o, \{\mathtt{read}\}), f_s, f_o)) \\ \text{if } \ R = \langle +, \{s\}, o, \mathtt{read}\rangle \\ \wedge\ f_o(o) \preceq f_s(s) \wedge \{o' \in \mathcal{O} \mid (s, o', \mathtt{write}) \Subset m\ \wedge \neg(f_o(o') \preceq f_o(o))\} = \emptyset \\ \\ (\mathsf{yes}, (m \oplus (\{s\}, o, \{\mathtt{write}\}), f_s, f_o)) \\ \text{if } \ R = \langle +, \{s\}, o, \mathtt{write}\rangle \\ \wedge\ \{o' \in \mathcal{O} \mid (s, o', \mathtt{read}) \Subset m\ \wedge \neg(f_o(o) \preceq f_o(o'))\} = \emptyset \\ \\ (\mathsf{no}, (m, f_s, f_o)) \text{ otherwise} \end{cases}$$

where

$$(s, o, x) \Subset m \Leftrightarrow \exists S \in \wp(\mathcal{S})\ \exists E \in \wp(\mathcal{A})\ (S, o, E) \in m \wedge s \in S \wedge x \in E$$

Furthermore, we introduce the following set of initial states:

$$\Sigma_I^{BLP} = \{\sigma \in \Sigma_{BLP} \mid \alpha(\sigma) = \emptyset\}$$

Clearly τ'_{BLP} is a restriction of τ_{BLP}, and it is easy to prove the following proposition.

Proposition 2. *$(\tau'_{BLP}, \Sigma_I^{BLP})$ is a $\mathbb{P}_{BLP}[\rho_{BLP}]$-complete non-discriminatory implementation such that $\mathbb{P}_{BLP}[\rho_{BLP}], [\![\mathcal{R}]\!]_{BLP} \vdash (\tau'_{BLP}, \Sigma_I^{BLP})$*

Implementation of the Chinese Wall policy Given a security parameter $\rho_{CW} = (\mathbb{C}, \mathbb{E}, f_{\mathbb{E}})$, a state $\sigma = (m, f_o) \in \Sigma_{CW}$, and a request $R \in \mathcal{R}$, we define τ_{CW} as follows

$$
\begin{aligned}
&\tau_{CW}(R, (m, f_o)) \\
&= \begin{cases}
(\mathsf{yes}, (m \oplus (\{s\}, o, \{\mathtt{read}\}), f_o)) \\
\text{if } R = \langle +, \{s\}, o, \mathtt{read}\rangle \wedge f_o(o) = (\bot, \ldots, \bot) \\
\\
(\mathsf{yes}, (m \oplus (\{s\}, o, \{\mathtt{read}\}), f_o)) \\
\text{if } \ R = \langle +, \{s\}, o, \mathtt{read}\rangle \wedge f_o(o) \neq (\bot, \ldots, \bot) \\
\wedge \left\{ o' \in \mathcal{O} \mid (s, o', \mathtt{read}) \Subset m \wedge \exists i \begin{pmatrix} \pi_i(f_o(o)) \neq \bot \ \wedge \ \pi_i(f_o(o')) \neq \bot \\ \wedge\, \pi_i(f_o(o)) \neq \pi_i(f_o(o')) \end{pmatrix} \right\} = \emptyset \\
\wedge \left\{ o'' \in \mathcal{O} \mid (s, o'', \mathtt{write}) \Subset m \wedge \exists i \begin{pmatrix} \pi_i(f_o(o)) \neq \bot \\ \wedge\, \pi_i(f_o(o)) \neq \pi_i(f_o(o'')) \end{pmatrix} \right\} = \emptyset \\
\\
(\mathsf{yes}, (m \oplus (\{s\}, o, \{\mathtt{write}\}), f_o)) \\
\text{if } \ R = \langle +, \{s\}, o, \mathtt{write}\rangle \\
\wedge \left\{ o' \in \mathcal{O} \mid (s, o', \mathtt{read}) \Subset m \ \wedge \ f_o(o') \neq (\bot, \ldots, \bot) \wedge \exists i \begin{pmatrix} \pi_i(f_o(o)) \neq \bot \\ \wedge\, \pi_i(f_o(o)) \neq \pi_i(f_o(o')) \end{pmatrix} \right\} = \emptyset \\
\wedge \left\{ o'' \in \mathcal{O} \mid (s, o'', \mathtt{write}) \Subset m \wedge \exists i \begin{pmatrix} \pi_i(f_o(o)) \neq \bot \ \wedge \ \pi_i(f_o(o'')) \neq \bot \\ \wedge\, \pi_i(f_o(o)) \neq \pi_i(f_o(o'')) \end{pmatrix} \right\} = \emptyset \\
\\
(\mathsf{no}, (m, f_o)) \text{ otherwise}
\end{cases}
\end{aligned}
$$

Furthermore, we introduce the following set of initial states:

$$\Sigma_I^{CW} = \{\sigma \in \Sigma_{CW} \mid \alpha(\sigma) = \emptyset\}$$

Here again, it is easy to prove the following proposition.

Proposition 3. *$(\tau_{CW}, \Sigma_I^{CW})$ is a $\mathbb{P}_{CW}[\rho_{CW}]$-complete non-discriminatory implementation such that $\mathbb{P}_{CW}[\rho_{CW}], [\![\mathcal{R}]\!]_{CW} \vdash (\tau_{CW}, \Sigma_I^{CW})$*

4.4 Comparing access control policies (2)

As we said, in order to be able to compare two access control policies $\mathbb{P}_1[\rho_1] = (\mathcal{S}, \mathcal{O}, \mathcal{A}, \Sigma_1, \Omega_1)$ and $\mathbb{P}_2[\rho_2] = (\mathcal{S}, \mathcal{O}, \mathcal{A}, \Sigma_2, \Omega_2)$ we have to introduce the usual definition of simulation relations between their implementations. Of course, this can be done only if implementations are defined over the same set of requests.

Definition 7 (Simulation of transition functions). *Given two transition functions $\tau_1 : \mathcal{R} \times \Sigma_1 \to \mathcal{D} \times \Sigma_1$ and $\tau_2 : \mathcal{R} \times \Sigma_2 \to \mathcal{D} \times \Sigma_2$, we say that τ_2*

simulates τ_1*, and in this case we write* $\tau_1 \overset{\kappa_\Sigma}{\preceq} \tau_2$*, iff there exists a binary relation* $\kappa_\Sigma \subseteq \Sigma_1 \times \Sigma_2$ *such that:*

$$\begin{array}{l} \forall \sigma_1, \sigma_1' \in \Sigma_1 \ \forall \sigma_2 \in \Sigma_2 \ \forall R \in \mathcal{R} \ \forall a \in \mathcal{D} \\ \quad ((\sigma_1, \sigma_2) \in \kappa_\Sigma \wedge \tau_1(R, \sigma_1) = (a, \sigma_1')) \\ \Rightarrow (\exists \sigma_2' \in \Sigma_2 \ (\sigma_1', \sigma_2') \in \kappa_\Sigma \wedge \tau_2(R, \sigma_2) = (a, \sigma_2')) \end{array}$$

We extend this definition to implementations of access control policies.

Definition 8 (Simulation of implementations). *We say that the implementation* (τ_2, Σ_I^2) *simulates* (τ_1, Σ_I^1)*, and in this case we write* $(\tau_1, \Sigma_I^1) \overset{\kappa_\Sigma}{\preceq} (\tau_2, \Sigma_I^2)$*, iff there exists a binary relation* $\kappa_\Sigma \subseteq \Sigma_1 \times \Sigma_2$ *such that:*

$$\begin{array}{l} \quad \tau_1 \overset{\kappa_\Sigma}{\preceq} \tau_2 \\ \wedge \ \forall \sigma_1 \in \Sigma_I^1 \ \exists \sigma_2 \in \Sigma_I^2 \quad (\sigma_1, \sigma_2) \in \kappa_\Sigma \\ \wedge \ \forall (\sigma_1, \sigma_2) \in \kappa_\Sigma \ \alpha(\sigma_1) = \alpha(\sigma_2) \end{array}$$

These definitions allow to easily prove the following lemma.

Lemma 2. *Let* (τ_1, Σ_I^1) *and* (τ_2, Σ_I^2) *be two implementations and* κ_Σ *be a simulation relation such that* $(\tau_1, \Sigma_I^1) \overset{\kappa_\Sigma}{\preceq} (\tau_2, \Sigma_I^2)$*. We have:*

$$\forall \sigma_1 \in \Gamma_{\tau_1}(\Sigma_I^1) \ \exists \sigma_2 \in \Gamma_{\tau_2}(\Sigma_I^2) \ (\sigma_1, \sigma_2) \in \kappa_\Sigma$$

We are now in position to express in a formal way that a policy $\mathbb{P}_1[\rho_1]$ is more restrictive than a policy $\mathbb{P}_2[\rho_2]$ as suggested in subsection 3.3.

Definition 9 (Comparison of access control policies). *Given two access control policies* $\mathbb{P}_1[\rho_1] = (\mathcal{S}, \mathcal{O}, \mathcal{A}, \Sigma_1, \Omega_1)$ *and* $\mathbb{P}_2[\rho_2] = (\mathcal{S}, \mathcal{O}, \mathcal{A}, \Sigma_2, \Omega_2)$*, a set of requests* $\mathcal{R}$ *together with semantics* $[\![\mathcal{R}]\!]_{\Sigma_1}$ *and* $[\![\mathcal{R}]\!]_{\Sigma_2}$*,* $\mathbb{P}_1[\rho_1] \trianglelefteq \mathbb{P}_2[\rho_2]$ *iff:*

$$\begin{array}{l} \forall \tau_1 : \mathcal{R} \times \Sigma_1 \to \mathcal{D} \times \Sigma_1 \ \forall \Sigma_I^1 \subseteq \Sigma_1 \\ \quad \mathbb{P}_1[\rho_1], [\![\mathcal{R}]\!]_{\Sigma_1} \vdash (\tau_1, \Sigma_I^1) \\ \Rightarrow \exists \tau_2 : \mathcal{R} \times \Sigma_2 \to \mathcal{D} \times \Sigma_2 \ \exists \Sigma_I^2 \subseteq \Sigma_2 \ \exists \kappa_\Sigma \subseteq \Sigma_1 \times \Sigma_2 \\ \quad (\mathbb{P}_2[\rho_2], [\![\mathcal{R}]\!]_{\Sigma_2} \vdash (\tau_2, \Sigma_I^2) \wedge (\tau_1, \Sigma_I^1) \overset{\kappa_\Sigma}{\preceq} (\tau_2, \Sigma_I^2)) \end{array}$$

Note that $\trianglelefteq$ is a preorder over access control policies.

As we said, many implementations exist for an access control policy and in order to prove that a policy is more restrictive than another one, we would like to restrict the number of implementations to consider. In practice, a possible way to achieve this goal is to define a mapping from Σ_1 to Σ_2 satisfying some good properties and allowing to define a left-total functional simulation relation κ_Σ.

Proposition 4.

$$\begin{array}{l}\forall \mathbb{P}_1[\rho_1]\ \mathbb{P}_2[\rho_2]\ \forall(\tau_1,\Sigma_I^1)\ (\tau_1',\Sigma_I^{'1})\ (\tau_2,\Sigma_I^2)\ \forall \kappa_\Sigma \subseteq \Sigma_{1\,|\Omega_1} \times \Sigma_{2\,|\Omega_2} \\ \left(\begin{array}{l} \kappa_\Sigma \text{ is left-total and functional} \\ \wedge\ [\![\mathcal{R}]\!]_{\Sigma_1}, \mathbb{P}_1[\rho_1] \vdash (\tau_1,\Sigma_I^1),(\tau_1',\Sigma_I^{'1}) \\ \wedge\ [\![\mathcal{R}]\!]_{\Sigma_2}, \mathbb{P}_2[\rho_2] \vdash (\tau_2,\Sigma_I^2) \\ \wedge\ \tau_1 \text{ is } \equiv_{\kappa_\Sigma}\text{-preserving} \\ \wedge\ (\tau_1',\Sigma_I^{'1}) \sqsubseteq_{\Gamma,\equiv_{\kappa_\Sigma}} (\tau_1,\Sigma_I^1) \wedge (\tau_1,\Sigma_I^1) \overset{\kappa_\Sigma}{\preceq} (\tau_2,\Sigma_I^2) \end{array}\right) \\ \Rightarrow \exists(\tau_2',\Sigma_I^{'2}) \\ \quad \Big([\![\mathcal{R}]\!]_{\Sigma_2}, \mathbb{P}_2[\rho_2] \vdash (\tau_2',\Sigma_I^{'2}) \wedge (\tau_1',\Sigma_I^{'1}) \overset{\kappa_\Sigma}{\preceq} (\tau_2',\Sigma_I^{'2}) \wedge (\tau_2',\Sigma_I^{'2}) \sqsubseteq_{\Gamma,=} (\tau_2,\Sigma_I^2)\Big)\end{array}$$

where $\equiv_{\kappa_\Sigma}$ *is defined as follows:*

$$\forall \sigma,\sigma' \in \Sigma_1\ \sigma \equiv_{\kappa_\Sigma} \sigma' \Leftrightarrow \exists \sigma_2 \in \Sigma_2\ ((\sigma,\sigma_2) \in \kappa_\Sigma \wedge (\sigma',\sigma_2) \in \kappa_\Sigma)$$

Proof. In order to prove the desired properties we prove the following statements.
• (–1–). First, let us define $(\tau_2', \Sigma_I^{'2})$. The set $\Sigma_I^{'2}$ can be defined as:

$$\Sigma_I^{'2} = \{\sigma_2 \in \Sigma_2 \mid \exists \sigma_1 \in \Sigma_I^{'1}\ (\sigma_1,\sigma_2) \in \kappa_\Sigma\}$$

Now, let σ_2 be a state in Σ_2 and R a request. Two cases are possible.

1. There does not exist a state $\sigma_1 \in \Sigma_1$ such that $(\sigma_1,\sigma_2) \in \kappa_\Sigma$ and we define $\tau_2'(R,\sigma_2)$ as (no,σ_2).
2. There exists a state $\sigma_1 \in \Sigma_1$ such that $(\sigma_1,\sigma_2) \in \kappa_\Sigma$. Let σ_1' be a state such that for an answer a, $\tau_1'(R,\sigma_1) = (a,\sigma_1')$. Since, by hypothesis, κ_Σ is left-total, there exists a state $\sigma_2' \in \Sigma_2$ such that $(\sigma_1',\sigma_2') \in \kappa_\Sigma$ and we define $\tau_2'(R,\sigma_2)$ as (a,σ_2').

• (–2–). We prove that: $\forall \sigma_2 \in \Gamma_{\tau_2'}(\Sigma_I^{'2})\ \exists \sigma_1 \in \Gamma_{\tau_1'}(\Sigma_I^{'1})\ (\sigma_1,\sigma_2) \in \kappa_\Sigma$
It suffices to prove by induction over n that:

$$\forall n \in \mathbb{N}\ \forall \sigma_2 \in \Gamma_{\tau_2'}^n(\Sigma_I^{'2})\ \exists \sigma_1 \in \Gamma_{\tau_1'}^n(\Sigma_I^{'1})\ (\sigma_1,\sigma_2) \in \kappa_\Sigma$$

If $n = 0$ then $\sigma_2 \in \Sigma_I^{'2}$ and we can conclude by definition of $\Sigma_I^{'2}$. If $n = k{+}1$, then there exists a state $\sigma_A \in \Gamma_{\tau_2'}^k(\Sigma_I^{'2})$ such that for a request R and an answer a, $\tau_2'(R,\sigma_A) = (a,\sigma_2)$. By induction hypothesis, there exists a state $\sigma_B \in \Gamma_{\tau_1'}^k(\Sigma_I^{'1})$ such that $(\sigma_B,\sigma_A) \in \kappa_\Sigma$. Let σ_C be a state such that $\tau_1'(R,\sigma_B) = (a',\sigma_C)$. Furthermore, by construction of τ_2', there exists a state $\sigma_D \in \Sigma_1$ such that $(\sigma_D,\sigma_A) \in \kappa_\Sigma$, $\tau_1'(R,\sigma_D) = (a,\sigma_1)$ and $(\sigma_1,\sigma_2) \in \kappa_\Sigma$. Moreover, by definition of $\equiv_{\kappa_\Sigma}$, we have $\sigma_B \equiv_{\kappa_\Sigma} \sigma_D$. We only know that the state σ_C is in $\Gamma_{\tau_1'}^k(\Sigma_I^{'1})$ and we to prove that $(\sigma_C,\sigma_2) \in \kappa_\Sigma$. Indeed, since $(\tau_1',\Sigma_I^{'1}) \sqsubseteq_{\Gamma,\equiv_{\kappa_\Sigma}} (\tau_1,\Sigma_I^1)$ and, by hypothesis, τ_1 is $\equiv_{\kappa_\Sigma}$-preserving, by lemma 1, τ_1' is also $\equiv_{\kappa_\Sigma}$-preserving, and we obtain $\sigma_1 \equiv_{\kappa_\Sigma} \sigma_C$. Now we can conclude from $(\sigma_1,\sigma_2) \in \kappa_\Sigma$ and by definition of $\equiv_{\kappa_\Sigma}$.
• (–3–). Now we prove that $\Gamma_{\tau_2'}(\Sigma_I^{'2}) \subseteq \Gamma_{\tau_2}(\Sigma_I^2)$. Let σ_2 be a state in $\Gamma_{\tau_2'}(\Sigma_I^{'2})$, then, from the step (–2–) of this proof we can conclude that there exists a state

$\sigma_1 \in \Gamma_{\tau_1'}(\Sigma_I'^1)$ such that $(\sigma_1, \sigma_2) \in \kappa_\Sigma$. Since we have $(\tau_1', \Sigma_I'^1) \sqsubseteq_\Gamma (\tau_1, \Sigma_I^1)$, it follows $\sigma_1 \in \Gamma_{\tau_1}(\Sigma_I^1)$. Now, since $(\tau_1, \Sigma_I^1) \overset{\kappa_\Sigma}{\sim} (\tau_2, \Sigma_I^2)$, by lemma 2, there exists a state $\sigma_2' \in \Gamma_{\tau_2}(\Sigma_I^2)$ such that $(\sigma_1, \sigma_2') \in \kappa_\Sigma$. Now, since κ_Σ is functional, we obtain $\sigma_2 = \sigma_2'$ and we can conclude.

• (–4–). Let us prove that $(\tau_1', \Sigma_I'^1) \overset{\kappa_\Sigma}{\sim} (\tau_2', \Sigma_I'^2)$. Let us first prove that for all $\sigma_1 \in \Sigma_I'^1$, there exists a state $\sigma_2 \in \Sigma_I'^2$ such that $(\sigma_1, \sigma_2) \in \kappa_\Sigma$. Indeed, since κ_Σ is left-total, if $\sigma_1 \in \Sigma_I'^1 \subseteq \Sigma_1$, then there exists a state $\sigma_2 \in \Sigma_2$ such that $(\sigma_1, \sigma_2) \in \kappa_\Sigma$ and we can conclude by definition of $\Sigma_I'^2$. Now, let us prove that $\tau_1' \overset{\kappa_\Sigma}{\sim} \tau_2'$. Let $\sigma_A, \sigma_B \in \Sigma_1$, $R \in \mathcal{R}$ and $a \in \mathcal{D}$ such that $\tau_1'(R, \sigma_A) = (a, \sigma_B)$ and $\sigma_C \in \Sigma_2$ such that $(\sigma_A, \sigma_C) \in \kappa_\Sigma$. By construction of τ_2', there exists a state $\sigma_E \in \Sigma_1$ such that $(\sigma_E, \sigma_C) \in \kappa_\Sigma$, and a state $\sigma_F \in \Sigma_1$ such that $\tau_1'(R, \sigma_E) = (a', \sigma_F)$. Moreover $\tau_2'(R, \sigma_C) = (a', \sigma_D)$ for a state $\sigma_D \in \Sigma_2$ such that $(\sigma_F, \sigma_D) \in \kappa_\Sigma$. Since, by hypothesis, $(\tau_1', \Sigma_I'^1) \sqsubseteq_{\Gamma, \equiv_{\kappa_\Sigma}} (\tau_1, \Sigma_I^1)$ and, τ_1 is $\equiv_{\kappa_\Sigma}$-preserving, by lemma 1, τ_1' is also $\equiv_{\kappa_\Sigma}$-preserving, Now, by definition of $\equiv_{\kappa_\Sigma}$, we have $\sigma_A \equiv_{\kappa_\Sigma} \sigma_E$, and then, $a = a'$ and $\sigma_B \equiv_{\kappa_\Sigma} \sigma_F$. Now we can conclude from $(\sigma_F, \sigma_D) \in \kappa_\Sigma$ and by definition of $\equiv_{\kappa_\Sigma}$.

• (–5–). Now, let us prove that $(\tau_2', \Sigma_I'^2) \sqsubseteq_{\Gamma, =} (\tau_2, \Sigma_I^2)$. We have already proved that $\Gamma_{\tau_2'}(\Sigma_I'^2) \subseteq \Gamma_{\tau_2}(\Sigma_I^2)$, so we have now to prove that $\tau_2' \sqsubseteq_= \tau_2$. Any transition function is clearly $=$-preserving, so τ_2' is $=_{\mathsf{no}}$-preserving. Now, let σ_A, σ_B be two states in Σ_2, R a request such that $\tau_2'(R, \sigma_A) = (\mathsf{yes}, \sigma_B)$. Let us prove that $\tau_2(R, \sigma_A) = (\mathsf{yes}, \sigma_B)$. Indeed, by definition of τ_2', there exist two states σ_C and σ_D in Σ_1 such that $(\sigma_C, \sigma_A) \in \kappa_\Sigma$, $(\sigma_D, \sigma_B) \in \kappa_\Sigma$ and $\tau_1'(R, \sigma_C) = (\mathsf{yes}, \sigma_D)$. Since $(\tau_1', \Sigma_I'^1) \sqsubseteq_{\Gamma, \equiv_{\kappa_\Sigma}} (\tau_1, \Sigma_I^1)$, there exists a state $\sigma_E \in \Sigma_1$ such that $\tau_1(R, \sigma_C) = (\mathsf{yes}, \sigma_E)$ and $\sigma_D \equiv_{\kappa_\Sigma} \sigma_E$. From $(\tau_1, \Sigma_I^1) \overset{\kappa_\Sigma}{\sim} (\tau_2, \Sigma_I^2)$, we know that there exists a state σ_F in Σ_2 such that $\tau_2(R, \sigma_A) = (\mathsf{yes}, \sigma_F)$ and $(\sigma_E, \sigma_F) \in \kappa_\Sigma$. Now, since, by hypothesis, κ_Σ is functional, we obtain $\sigma_B = \sigma_F$, and then we can conclude.

• (–6–). At last, we are now in position to prove that $[\![\mathcal{R}]\!]_{\Sigma_2}, \mathbb{P}_2[\rho_2] \vdash (\tau_2', \Sigma_I'^2)$. Indeed, since $\mathbb{P}_2[\rho_2] \vdash (\tau_2, \Sigma_I^2)$ and $\Gamma_{\tau_2'}(\Sigma_I'^2) \subseteq \Gamma_{\tau_2}(\Sigma_I^2)$, then we have $\Gamma_{\tau_2'}(\Sigma_I'^2) \subseteq \Sigma_{2|\Omega_2}$. Furthermore, $[\![\mathcal{R}]\!]_{\Sigma_2} \vdash (\tau_2', \Sigma_I'^2)$ follows from the fact that $[\![\mathcal{R}]\!]_{\Sigma_2} \vdash (\tau_2, \Sigma_I^2)$ and $\tau_2' \sqsubseteq_= \tau_2$.

This proposition allows to only consider $\sqsubseteq_{\Gamma, \equiv_{\kappa_\Sigma}}$-maximal implementations[1] and to show that all these implementations can be simulated.

4.5 Application

This subsection shows the practical applicability of our framework. Recall that in subsection 4.3, we have defined correct classical implementations $(\tau_{BLP}', \Sigma_I^{BLP})$ and $(\tau_{CW}, \Sigma_I^{CW})$ of $\mathbb{P}_{BLP}[\rho_{BLP}]$ and $\mathbb{P}_{CW}[\rho_{CW}]$.

Now, we would like to prove that the Chinese Wall policy is more restrictive than the Bell and LaPadula policy. Hence, we have to consider simulations of

[1] Implementations (τ, Σ_I) such that there does not exist any other implementation (τ', Σ_I') with $(\tau, \Sigma_I) \sqsubseteq_{\Gamma, \equiv} (\tau', \Sigma_I')$.

implementations of $\mathbb{P}_{CW}[\rho_{CW}]$ by implementations of the Bell and LaPadula policy. However, we have to define the security parameters ρ_{BLP} from ρ_{CW}. To achieve this goal, we define a lattice-based interpretation of the Chinese Wall policy. This interpretation is similar to the one introduced in [15, 16], however our interpretation, done in a formal setting, does not require notions like users or principals. We first define a lattice of security levels according to a set of conflict-of-interest classes and a set of companies:

$$\mathcal{L}_{CW} = \prod_{i=1}^{n} (\{\top\} \cup f_{\mathbb{E}}(c_i) \cup \{\bot\})$$

The order relation $\preceq$ over $\mathcal{L}_{CW}$ is defined as follows :

$$\begin{array}{l} \forall x_1, x_2 \in \mathcal{L}_{CW}\ x_1 = (e_1, \cdots, e_n) \preceq x_2 = (e'_1, \cdots, e'_n) \\ \Leftrightarrow \forall i\ (1 \leq i \leq n)\ e_i = \bot \vee e_i = e'_i \vee e'_i = \top \end{array}$$

For example, figure 1 illustrates such a lattice when $\mathbb{C} = \{c_1, c_2\}$, $f_{\mathbb{E}}(c_1) = \{d_{11}, d_{12}\}$ and $f_{\mathbb{E}}(c_2) = \{d_{21}, d_{22}\}$. When defining a lattice-based interpretation of the Chinese Wall policy, we will define security functions f_s and f_o:

$$\begin{array}{l} f_s : \mathcal{S} \rightarrow \prod_{i=1}^{n} (\{\top\} \cup f_{\mathbb{E}}(c_i)) \\ f_o : \mathcal{O} \rightarrow \prod_{i=1}^{n} (f_{\mathbb{E}}(c_i) \cup \{\bot\}) \end{array}$$

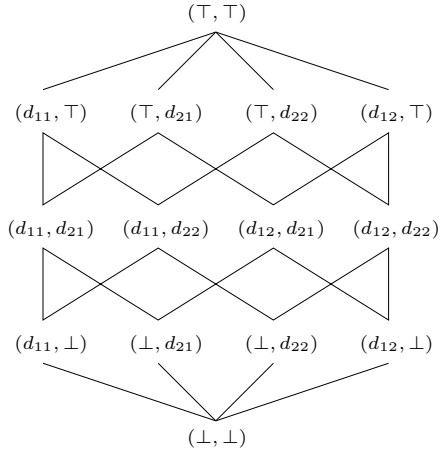


Fig. 1. Example of lattice of security levels

We are now in position to prove $\mathbb{P}_{CW}[\rho_{CW}] \trianglelefteq \mathbb{P}_{BLP}[\mathcal{L}_{CW}]$. So, we have to prove that every implementation of $\mathbb{P}_{CW}[\rho_{CW}]$ can be simulated by an implementation of $\mathbb{P}_{BLP}[\mathcal{L}_{CW}]$. The lattice-based interpretation introduced above allows

to define the relation $\kappa_\Sigma \subseteq \Sigma_{CW} \times \Sigma_{BLP}$:

$$\forall \sigma_1 = (m_1, f_o^1) \in \Sigma_{CW}\ \forall \sigma_2 = (m_2, f_s^2, f_o^2) \in \Sigma_{BLP}$$
$$(\sigma_1, \sigma_2) \in \kappa_\Sigma \Leftrightarrow \left(\begin{array}{l} \quad m_1 = m_2 \\ \wedge\ \forall o \in \mathcal{O}\ f_o^1(o) = f_o^2(o) \\ \wedge\ \forall s \in \mathcal{S}\ \forall o \in \mathcal{O}\ \forall x \in \mathcal{A} \\ \quad (s, o, x) \Subset m_2 \Rightarrow \\ \quad \forall i\ ((\pi_i(f_o^1(o)) = \bot \wedge \pi_i(f_s^2(s)) = \top) \\ \qquad \vee (\pi_i(f_o^1(o)) \neq \bot \wedge \pi_i(f_s^2(s)) = \pi_i(f_o^1(o)))) \end{array} \right)$$

Note that we can prove:

$$\forall \sigma_1 \in \Sigma_{CW}\ \forall \sigma_2 \in \Sigma_{BLP} \quad ((\sigma_1, \sigma_2) \in \kappa_\Sigma \wedge \Omega_{CW}(\sigma_1)) \Rightarrow \Omega_{BLP}(\sigma_2)$$

Hence, clearly, the restriction $\kappa_\Sigma \subseteq \Sigma_{CW\,|\Omega_{CW}} \times \Sigma_{BLP\,|\Omega_{BLP}}$ defines a left-total functional relation. Now, proposition 4 allows to prove that $\mathbb{P}_{CW}[\rho_{CW}] \trianglelefteq \mathbb{P}_{BLP}[\mathcal{L}_{CW}]$ since we can define a correct implementation $(\tau''_{BLP}, \Sigma_I^{''BLP})$ such that κ_Σ is a simulation relation between $(\tau_{CW}, \Sigma_I^{CW})$ and $(\tau''_{BLP}, \Sigma_I^{''BLP})$ and we can prove that $(\tau_{CW}, \Sigma_I^{CW})$ is a maximal implementation of the Chinese Wall policy.

The implementation $(\tau''_{BLP}, \Sigma_I^{''BLP})$ can be defined as follows. First we introduce the set of initial states:

$$\Sigma_I^{''BLP} = \left\{ \begin{array}{c} \sigma \in \Sigma_{BLP} \mid \alpha(\sigma) = \emptyset \wedge \forall s \in \mathcal{S}\ f_s(s) = (\top, \cdots, \top) \\ \wedge \exists \sigma' \in \Sigma_I^{CW}\ (\sigma', \sigma) \in \kappa_\Sigma \end{array} \right\}$$

Then, we define the transition function:

$$\tau''_{CW}(R, (m, f_s, f_o))$$

$$= \left\{ \begin{array}{l} (\mathsf{yes}, (m \oplus (\{s\}, o, \{\mathtt{read}\}), f_s \circledast (s, o), f_o)) \\ \text{if}\ \ R = \langle +, \{s\}, o, \mathtt{read} \rangle \\ \wedge\ \ f_o(o) \preceq f_s(s) \wedge \{o' \in \mathcal{O} \mid (s, o', \mathtt{write}) \Subset m\ \ \wedge \neg(f_o(o') \preceq f_o(o))\} = \emptyset \\ \\ (\mathsf{yes}, (m \oplus (\{s\}, o, \{\mathtt{write}\}), f_s \circledast (s, o), f_o)) \\ \text{if}\ \ R = \langle +, \{s\}, o, \mathtt{write} \rangle \\ \wedge\ \ \{o' \in \mathcal{O} \mid (s, o', \mathtt{read}) \Subset m\ \ \wedge \neg(f_o(o) \preceq f_o(o'))\} = \emptyset \\ \\ (\mathsf{no}, (m, f_s, f_o)) \quad \text{otherwise} \end{array} \right.$$

with

$$\forall s' \in \mathcal{S}\ (f_s \circledast (s, o))(s') = \left\{ \begin{array}{ll} f_s(s') & \text{if } s \neq s' \\ (e_1, e_2, \ldots, e_n) & \text{if } s = s' \text{ with} \\ & e_i = \left\{ \begin{array}{ll} \pi_i(f_s(s)) & \text{if } \pi_i(f_o(o)) = \bot \\ \pi_i(f_o(o)) & \text{otherwise} \end{array} \right. \end{array} \right.$$

Note that τ''_{BLP} corresponds to τ'_{BLP} except that it changes security levels of subjects. Of course, here again, it is easy to prove the proposition:

Proposition 5. *$(\tau''_{BLP}, \Sigma_I^{''BLP})$ is a $\mathbb{P}_{BLP}[\rho_{BLP}]$-complete non-discriminatory implementation such that $\mathbb{P}_{BLP}[\rho_{BLP}], [\![\mathcal{R}]\!]_{BLP} \vdash (\tau''_{BLP}, \Sigma_I^{''BLP})$*

Last, the two following propositions allow to prove that $\mathbb{P}_{CW}[\rho_{CW}] \trianglelefteq \mathbb{P}_{BLP}[\mathcal{L}_{CW}]$.

Proposition 6. $(\tau_{CW}, \Sigma_I^{CW}) \overset{\kappa_\Sigma}{\preceq} (\tau''_{BLP}, \Sigma_I^{''BLP})$

Proposition 7.

$$\forall (\tau'_{CW}, \Sigma_I^{'CW})$$
$$[\![\mathcal{R}]\!]_{\Sigma_{CW}}, \mathbb{P}_{CW}[\rho_{CW}] \vdash (\tau'_{CW}, \Sigma_I^{'CW}) \Rightarrow (\tau'_{CW}, \Sigma_I^{'CW}) \sqsubseteq_{\Gamma,=} (\tau_{CW}, \Sigma_I^{CW})$$

It is also possible to prove that $\mathbb{P}_{BLP}[\rho_{BLP}] \not\trianglelefteq \mathbb{P}_{CW}[\rho_{CW}]$. Indeed, let us show that $(\tau'_{BLP}, \Sigma_I^{'BLP})$ cannot be simulated by a non-discriminatory implementation of the Chinese Wall policy. Let us suppose that there exist a simulation relation κ_Σ and a "correct" non-discriminatory implementation $(\tau'_{CW}, \Sigma_I^{'CW})$ such that

$$(\tau'_{BLP}, \Sigma_I^{'BLP}) \overset{\kappa_\Sigma}{\preceq} (\tau'_{CW}, \Sigma_I^{'CW})$$

and then let us show that we get a contradiction.

Let us define the lattice $\mathcal{L} = \{\bot, \top\}$ with $\bot \prec \top$, the set of subjects $\mathcal{S} = \{s_1, s_2\}$ and the set of objects $\mathcal{O} = \{o_1, o_2, o_3\}$. We then consider the functions f_s and f_o such that $f_s(s_1) = f_s(s_2) = f_o(o_1) = f_o(o_2) = \bot$ and $f_o(o_3) = \top$. Let m be the following set of current access:

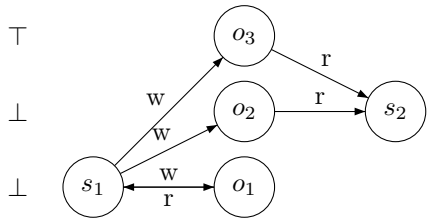
$$\{(\{s_1\}, o_1, \{\texttt{read}\}), (\{s_1\}, o_1, \{\texttt{write}\}), (\{s_1\}, o_2, \{\texttt{write}\}), (\{s_1\}, o_3, \{\texttt{write}\})\}$$

and $\sigma_1 \in \Sigma_{BLP}$ be the state (m, f_s, f_o). Clearly, we have $\Omega_{BLP}(\sigma_1)$. Since κ_Σ has to preserve the set of current access, defining a state $\sigma_2 = (m, f_o^2) \in \Sigma_{CW}$ such that $(\sigma_1, \sigma_2) \in \kappa_\Sigma$ requires that $f_o^2(o_1) = f_o^2(o_2) = f_o^2(o_3)$ and then $\iota_O(\sigma_2, o_2) = \iota_O(\sigma_2, o_3)$ (if a subject is writing in two different objects, both have to be in the same company). Now, suppose that s_2 asks to read o_2. Since $f_o(o_2) \preceq f_s(s_2)$ we have:

$$\tau'_{BLP}(\langle +, \{s_2\}, o_2, \texttt{read}\rangle, \sigma_1) = (\mathsf{yes}, \sigma_3)$$

where σ_3 is such that $\alpha(\sigma_3) = m \oplus (\{s_2\}, o_2, \{\texttt{read}\})$ and $\gamma(\sigma_3) = \gamma(\sigma_1)$. However, if s_2 asks to read o_3, this has to be denied by τ'_{BLP} because $\neg(f_o(o_3) \preceq f_s(s_2))$, and it follows:

$$\tau'_{BLP}(\langle +, \{s_2\}, o_3, \texttt{read}\rangle, \sigma_1) = (\mathsf{no}, \sigma_1)$$



Since we only consider non-discriminatory implementations, τ'_{CW} must allow the same actions for o_2 and o_3 from the state σ_2 and then:

$$\tau'_{CW}(\langle +, \{s_2\}, o_2, \texttt{read}\rangle, \sigma_2) = \tau'_{CW}(\langle +, \{s_2\}, o_3, \texttt{read}\rangle, \sigma_2)$$

which induces a contradiction.

5 Conclusion

In this paper, we have introduced a formalism allowing to specify and to define access control policies and their implementations at several levels of specification. Such a formalism also allows to reason over security policies and to compare them. Indeed, in this paper, we have introduced a way to compare access control policies based on the notion of simulation of implementations. While from a theoretical point of view, this allows to define a preorder relation over access control policies, from a more practical point of view, this allows to consider implementations by reusing. Furthermore, we think that our work could be a first step to consider composition of policies.

Of course, since this paper does not introduce any new access control policy and deals with well-known concepts, one could think that it does not contain any new material. However, our aim was to describe access control policies at a level of formalisation that is very close to a "computer-assisted formal specification" and/or implementation. Hence, one of our motivations was to fill the gap between (in)formal papers and formal implementations in order to ease the development of a formal library of access control policies and to avoid some bugs (as shown in section 2). We think that these bugs may be dangerous because the engineers and scientists tend to trust the answers given by security systems, and the consequences of a wrong computation can range from a few days of time lost in tracking down the error, to a complete failure of the system designed by the engineers. Therefore, we think that the need of formalisation is strong if we want to reach high levels of safety.

As we said, access control software must be based on a security policy model and flaws in them may come from some incoherences in the policy model but

also from inconsistencies between the model and the code. Hence, our current works consist in implementing our generic framework within Focal (allowing to write programs, properties and proofs at the same level) and in instantiating such a framework with variants of classical access control policies, like Chinese Wall or RBAC. Actually, numerous variations of the same policies have been published, the exact meaning of such policies is not always clear and we would like to obtain a formal library of access control policies.

Acknowledgements Many thanks to Thérèse Hardin for enlightening discussions on this subject. We also thank anonymous referees for their very usefull remarks.

References

1. P. Amey. Dear sir, yours faithfully: an everyday story of formality. In *Practical Elements of Safety, Proceedings of the Twelfth Safety-critical Systems Symposium*. Spinger-Verlag, 2004.
2. D. Bell and L. LaPadula. Secure Computer Systems: a Mathematical Model. Technical Report MTR-2547 (Vol. II), MITRE Corp., Bedford, MA, May 1973.
3. J. Blond and C. Morisset. Formalisation et implantation d'une politique de sécurité d'une base de données. In INRIA, editor, *17ème Journées Francophones des Langages Applicatifs, JFLA'2006*, pages 71–86, 2006.
4. D. F. C. Brewer and M. J. Nash. The chinese wall security policy. In *Proc. IEEE Symposium on Security and Privacy*, pages 206–214, 1989.
5. A. Chander, J.C. Mitchell, and D. Dean. A state-transition model of trust management and access control. In *Proceedings of the 14th IEEE Computer Security Foundations Workshop CSFW*, pages 27–43. IEEE Computer Society Press, 2001.
6. C. Dubois, T. Hardin, and V. Viguié Donzeau Gouge. Building certified components within focal. In *Symposium on Trends in Functional Programming*, 2004.
7. E. Gureghian, Th. Hardin, and M. Jaume. A full formalisation of the Bell and Lapadula security model. Technical Report 2003-007, Univ. Paris 6, LIP6, 2003.
8. M. Jaume and C. Morisset. Formalisation and implementation of access control models. In *Information Assurance and Security (IAS'05) International Conference on Information Technology, ITCC*, pages 703–708. IEEE CS Press, 2005.
9. M. Jaume and C. Morisset. A formal approach to implement access control model. *Journal of Information assurance and security*, To appear.
10. L.J. LaPadula and D.E. Bell. Secure Computer Systems: A Mathematical Model. *Journal of Computer Security*, 4:239–263, 1996.
11. McLean. The algebra of security. In *Proc. IEEE Symposium on Security and Privacy*, pages 2–7. IEEE Computer Society Press, 1988.
12. OCaml Development Team. *OCAML Reference Manual Version 3.08*. INRIA-Rocquencourt, 2004.
13. Focal project. *Focal, version 0.2 Tutorial and reference manual*. LIP6 – INRIA – CNAM, sept 2004. Distribution available at: `http://focal.inria.fr`.
14. Logical Project. *The Coq Proof Assistant Reference Manual*. INRIA-Rocquencourt, 2002.
15. R. S. Sandhu. A lattice interpretation of the chinese wall policy. In *Proc. 15th NIST-NCSC National Computer Security Conference*, pages 329–339, 1992.
16. R. S. Sandhu. Lattice-Based Access Control Models. *IEEE Computer*, 26(11):9–19, November 1993.