Semantic comparison of security policies : from access control policies to flow properties

Mathieu Jaume
SPI LIP6
University Pierre and Marie Curie
Paris, France
Email: Mathieu.Jaume@lip6.fr

***Abstract*—This paper introduces two generic mechanisms allowing to compare security policies from a semantical point of view. First, a notion of embedding is defined in order to compare policies over a common domain. Then, interpretations of security policies are introduced in order to consider their properties over arbitrary domains. Thus, combining interpretations and embeddings allows to compare policies expressed over different domains. Along the lines of this paper, we illustrate our definitions by defining a flow-based interpretation of access control and by comparing classical access control policies according to a hierarchy of abstract flow policies, thus characterizing flow properties which can be ensured by access control policies.**

I. Introduction

Several points of view exist on security policies, among which two main approaches can be distinguished: the rule-based approach (which consists in specifying the conditions under which an action is granted) and the property-based approach (which consists in specifying the security properties the policy aims to enforce). In [1], these two approaches have been fully formalized and their expressive power has been compared. Moreover, an operational mechanism allowing to enforce such policies over transition systems has also been defined together with a formal proof of its soundness [2]. In fact, such a framework allows to characterize the various entities involved in the definition of a security policy together with their role. Hence, it provides a semantical specification of security policies. Several developments on access control policies and flow policies have been done within this framework. In [3], an operational mechanism (parameterized by an access control policy) allowing to detect illegal information flows has been formalized. Such a mechanism is useful when the access control mechanism is not sufficient to ensure flow properties. For example, this is generally the case for discretionary access control policies, while MLS (MultiLevel Security) policies are clearly access control policies that can be used to enforce some flow policies. In [4], by considering morphisms between terms algebras, a first formalization of the comparison between access control policies and flow policies has been proposed within a variant of our framework based on rewrite systems. However, such a work is strongly related to the notion of terms algebras and we aim to provide semantical comparison mechanisms which do not depend on the syntax used to describe the policies. Hence, in this paper, we investigate at a deeper and more abstract level the property-based approach in order to define generic comparison mechanisms of security policies from a semantical point of view. First, we introduce a notion of embeddings of security policies allowing to compare the expressive power of policies over the same domain. For example, this can be useful when considering two access control policies based on different notions of security configurations: for instance, this allows to know if authorizations induced by a hierarchy of roles for a RBAC (Role-Based Access Control) policy can be obtained with a partial order of security levels for a MLS policy. However, in practice, it can also be useful to compare policies over different domains. As a typical example, some access control policies are designed for ensuring flow properties: such policies do not deal with information flow but only with objects containing information to be traced. Characterizing which flow properties are ensured by which access control policies requires to interpret accesses by flows. Hence, we also introduce the generic notion of interpretation of security policies allowing to understand security properties expressed over some entities induced by a policy whose domain is based on different entities. In fact, combining embeddings and interpretations provides a powerful way to analyse and to compare security policies from a semantical point of view. Along the lines of this paper, we illustrate our definitions by considering access control policies and flow policies. First, we use embeddings to compare the expressive power of access control policies and to build a hierarchy of abstract flow policies. Then, by introducing interpretations, we add access control policies into the hierarchy of flow policies, thus providing a formal characterization of flow properties induced by classical access control policies.

II. Security policies

A. Property-based security policies

By following a property-based approach, a security policy is a characterization of secure elements of a set according to some security information. Thus, specifying a policy $\mathbb{P}$ first consists in defining a set $\mathbb{T}$ of "things" that the policy aims to control, called the security targets, in order to ensure the desired security properties (these "things" can be the actions simultaneously done in the system, or some information about the entities of the system). Then, a set $\mathcal{C}$ of security configurations is introduced: configurations correspond to the information needed to characterize secure elements of $\mathbb{T}$

DOI 10.1109/SPW.2012.33

according to the policy. Last, the policy is specified by a binary relation $\Vdash$ between targets and configurations: $c \Vdash t$ means that the target t is secure according to the configuration c.

Definition 1 (Security policies): A security policy is a tuple $\mathbb{P} = (\mathbb{T}, \mathcal{C}, \Vdash)$ where $\mathbb{T}$ is a set of security targets, $\mathcal{C}$ is a set of security configurations and $\Vdash \subseteq \mathcal{C} \times \mathbb{T}$ is a relation specifying secure targets according to configurations.

In fact, a policy can be viewed as the definition of a semantics for its configurations. Indeed, each configuration c denotes the set of targets:

$$[\![c]\!]_{\mathbb{P}} = \{t \in \mathbb{T} \mid c \Vdash t\}$$

that c authorizes. Such definition is similar to the one introduced in [5], in the context of access control, where a policy is defined in terms of sets of authorized accesses.

B. Access control policies

Access control policies aim at controlling sets of accesses that can be simultaneously done by active entities, the subjects in $\mathcal{S}$, over some passive entities, the objects in $\mathcal{O}$, according to some access modes belonging to a set $\mathcal{A}$ (in this paper, we consider the set $\mathcal{A} = \{\text{read}, \text{write}\}$). An access can be represented by a triple (s, o, a) expressing that the subject s has an access over the object o according to the access mode a. In this context, we define the set $\mathbb{T}_A = \wp(\mathcal{S} \times \mathcal{O} \times \mathcal{A})$ of security targets of access control policies as the powerset of the cartesian product $\mathcal{S} \times \mathcal{O} \times \mathcal{A}$: targets are sets of accesses. We consider here two main approaches for access control:

- HRU [6] and RBAC [7] policies which are based on permissions (respectively associated with subjects and roles),
- and MLS policies [8], like the well-known Bell & La-Padula [9] or Biba [10] policies, which are based on a partial order of security levels.

Formal definitions of these policies are summarized in table I.

A configuration for HRU is just a set $m \in \mathcal{C}_{\mathsf{hru}}$ of authorized accesses (both targets and configurations are sets of accesses and we have $\mathcal{C}_{\mathsf{hru}} = \mathbb{T}_A$), and the HRU policy specifies secure sets of accesses as sets only containing authorized accesses. Within the RBAC policy, each user of a system is associated with roles, themselves associated with elements of $\mathcal{P} = \mathcal{O} \times \mathcal{A}$, where $(o, a) \in \mathcal{P}$ specifies a permission to access to o according to a. Configurations are tuples:

$$(\mathsf{U}, (\mathsf{R}, \leq_{\mathsf{R}}), \mathsf{UA}, \mathsf{PA}, user, roles) \in \mathcal{C}_{\mathsf{rbac}}$$

where U is a set of users, $(\mathsf{R}, \leq_{\mathsf{R}})$ is the partial order of roles, $\mathsf{UA} \subseteq \mathsf{U} \times \mathsf{R}$ specifies which users can activate which roles (and the roles lower to them according to $\leq_{\mathsf{R}}$), $\mathsf{PA} \subseteq \mathcal{P} \times \mathsf{R}$ is a relation associating permissions with roles, $user : \mathcal{S} \to \mathsf{U}$ allows to know the user corresponding to a subject (viewed here as a session), and $roles : \mathcal{S} \to \wp(\mathsf{R})$ specifies the set of roles that have been activated by a subject, and such that $roles(s) \subseteq \mathsf{ER}(s)$ for all $s \in \mathcal{S}$, where, given a subject $s \in \mathcal{S}$, $\mathsf{ER}(s)$ is the set of roles that s can activate according to UA:

$$\mathsf{ER}(s) = \{r \in \mathsf{R} \mid \exists r' \in \mathsf{R} \;\; r \leq_{\mathsf{R}} r' \wedge \;\; (user(s), r') \in \mathsf{UA}\}$$

This allows to specifiy granted accesses as accesses done by subjects that have activated a role associated with the permissions of its accesses. Given a subject s, we write $\mathsf{EP}(s)$ the set of permissions associated with the roles activated by s:

$$\mathsf{EP}(s) = \bigcup_{r \in roles(s)} \left\{ \begin{array}{l} (o, a) \in \mathcal{P} \mid \\ \exists r' \in \mathsf{R} \; r' \leq_{\mathsf{R}} r \wedge ((o, a), r') \in \mathsf{PA} \end{array} \right\}$$

MLS policies are defined from a partially ordered set $(\mathcal{L}, \preceq)$ of security levels (or sensitivities) and consists in associating security levels with objects and/or subjects to specify what are authorized accesses in terms of security levels. We consider here four classical examples showing how to define confinment, confidentiality and integrity policies as MLS policies. Of course, the main purpose of these access control policies is to constrain information flows. We define two confinment policies, $\mathbb{P}^{\uparrow}_{\mathsf{mls}}$ ("no write down") and $\mathbb{P}^{\downarrow}_{\mathsf{mls}}$ ("no write up"), which are both based on the set $\mathcal{C}_{\mathsf{mls}} = \{(\mathcal{L}, \preceq, f_O)\}$ of configurations, where $f_O : \mathcal{O} \to \mathcal{L}$ defines the security level of objects. By considering configurations in $\mathcal{C}^S_{\mathsf{mls}} = \{(\mathcal{L}, \preceq, f_O, f_S)\}$ (where $f_S : \mathcal{S} \to \mathcal{L}$) specifying security levels to objects and subjects, it becomes possible to define the confidentiality policy $\mathbb{P}^C_{\mathsf{mls}}$ and the integrity policy $\mathbb{P}^I_{\mathsf{mls}}$.

C. Abstract flow policies

Flow policies aim at controlling information flows between entities of a system. This control is explicitly described by considering sets of flows occurring in a system, without any consideration about the origin of such flows (execution of a program, accesses done in a system, etc). Hence, such policies can be seen as abstract policies since we don't take here into account how such flows are generated. These flows can be flows between objects (confinment flow policy), flows from objects to subjects (confidentiality flow policy), and flows from subjects to objects (integrity flow policy). Hence, targets of such policies are tuple $(\overset{\mathsf{oo}}{\curvearrowright}, \overset{\mathsf{os}}{\curvearrowright}, \overset{\mathsf{so}}{\curvearrowright})$ where:

$$\overset{\mathsf{oo}}{\curvearrowright} \subseteq \overset{\mathsf{oo}}{\hookrightarrow} = \mathcal{O} \times \mathcal{O} \qquad \overset{\mathsf{os}}{\curvearrowright} \subseteq \overset{\mathsf{os}}{\hookrightarrow} = \mathcal{O} \times \mathcal{S} \qquad \overset{\mathsf{so}}{\curvearrowright} \subseteq \overset{\mathsf{so}}{\hookrightarrow} = \mathcal{S} \times \mathcal{O}$$

specifying three sets of flows done in the system: $o_1 \overset{\mathsf{oo}}{\curvearrowright} o_2$ means that the information contained into o_1 flows into o_2, $o \overset{\mathsf{os}}{\curvearrowright} s$ means that the subject s has (in a direct or indirect way) a read access over the information initially contained into o, and $s \overset{\mathsf{so}}{\curvearrowright} o$ means that the subject s has (in a direct or indirect way) a write access to o. Of course, targets specify "coherent" sets of flows and belong to the set:

$$\mathbb{T}_F = \left\{ \begin{array}{ll} \multicolumn{2}{l}{(\overset{\mathsf{oo}}{\curvearrowright}, \overset{\mathsf{os}}{\curvearrowright}, \overset{\mathsf{so}}{\curvearrowright}) \mid} \\ & \overset{\mathsf{oo}}{\curvearrowright} = \left(\overset{\mathsf{oo}}{\curvearrowright}\right)^{\star} \\ \wedge & \forall s \in \mathcal{S} \; \forall o_1, o_2 \in \mathcal{O} \\ & (o_1 \overset{\mathsf{oo}}{\curvearrowright} o_2 \wedge o_2 \overset{\mathsf{os}}{\curvearrowright} s) \Rightarrow o_1 \overset{\mathsf{os}}{\curvearrowright} s \\ \wedge & \forall s \in \mathcal{S} \; \forall o_1, o_2 \in \mathcal{O} \\ & (s \overset{\mathsf{so}}{\curvearrowright} o_1 \wedge o_1 \overset{\mathsf{oo}}{\curvearrowright} o_2) \Rightarrow s \overset{\mathsf{so}}{\curvearrowright} o_2 \\ \wedge & \overset{\mathsf{oo}}{\curvearrowright} = \left(\overset{\mathsf{so}}{\curvearrowright} \circ \overset{\mathsf{os}}{\curvearrowright}\right) \cup \{o \overset{\mathsf{oo}}{\hookrightarrow} o \mid o \in \mathcal{O}\} \end{array} \right\}$$

Indeed, the relation characterizing flows between objects is clearly transitive and if there is an information flow from o_1

TABLE I
ACCESS CONTROL POLICIES

Policy	Configuration	$c \Vdash E \Leftrightarrow$
$\mathbb{P}_{\mathsf{hru}}$	$m \in \mathcal{C}_{\mathsf{hru}}$	$E \subseteq m$
$\mathbb{P}_{\mathsf{rbac}}$	$(\mathsf{U}, (\mathsf{R}, \leq_{\mathsf{R}}), \mathsf{UA}, \mathsf{PA}, user, roles) \in \mathcal{C}_{\mathsf{rbac}}$	$\forall (s,o,a) \in \mathcal{S} \times \mathcal{O} \times \mathcal{A}\ (s,o,a) \in E \Rightarrow (o,a) \in \mathsf{EP}(s)$
$\mathbb{P}^{\uparrow}_{\mathsf{mls}}$	$(\mathcal{L}, \preceq, f_O) \in \mathcal{C}_{\mathsf{mls}}$	$\forall s \in \mathcal{S}\ \forall o_1, o_2 \in \mathcal{O}\ ((s, o_1, \text{read}) \in E \wedge (s, o_2, \text{write}) \in E) \Rightarrow f_O(o_1) \preceq f_O(o_2)$
$\mathbb{P}^{\downarrow}_{\mathsf{mls}}$	$(\mathcal{L}, \preceq, f_O) \in \mathcal{C}_{\mathsf{mls}}$	$\forall s \in \mathcal{S}\ \forall o_1, o_2 \in \mathcal{O}\ ((s, o_1, \text{read}) \in E \wedge (s, o_2, \text{write}) \in E) \Rightarrow f_O(o_2) \preceq f_O(o_1)$
$\mathbb{P}^{C}_{\mathsf{mls}}$	$(\mathcal{L}, \preceq, f_O, f_S) \in \mathcal{C}^{S}_{\mathsf{mls}}$	$\forall s \in \mathcal{S}\ \forall o_1, o_2 \in \mathcal{O}\ ((s, o_1, \text{read}) \in E \wedge (s, o_2, \text{write}) \in E) \Rightarrow f_O(o_1) \preceq f_O(o_2)$ $\wedge\ \forall (s, o, \text{read}) \in E\ f_O(o) \preceq f_S(s)$
$\mathbb{P}^{I}_{\mathsf{mls}}$	$(\mathcal{L}, \preceq, f_O, f_S) \in \mathcal{C}^{S}_{\mathsf{mls}}$	$\forall s \in \mathcal{S}\ \forall o_1, o_2 \in \mathcal{O}\ ((s, o_1, \text{read}) \in E \wedge (s, o_2, \text{write}) \in E) \Rightarrow f_O(o_2) \preceq f_O(o_1)$ $\wedge\ \forall (s, o, \text{write}) \in E\ f_O(o) \preceq f_S(s)$

to o_2 and if a subject s has a read access to o_2 (resp. has a write access to o_1), then s has also a read access to the information contained into o_1 (resp. has a write access to o_2). Furthermore, flows between objects are only generated by read and write accesses done by subjects over objects, and thus are obtained by composition[1] of flows between subjects and objects.

Flows policies are based on configurations specifying sets of authorized flows, from which we define several flow policies, summarized in table II. We write $\overset{\mathsf{oo}}{\rightsquigarrow}$, $\overset{\mathsf{os}}{\rightsquigarrow}$ and $\overset{\mathsf{so}}{\rightsquigarrow}$ the subsets of $\overset{\mathsf{oo}}{\hookrightarrow}$,$\overset{\mathsf{os}}{\hookrightarrow}$ and $\overset{\mathsf{so}}{\hookrightarrow}$ specifying authorized sets of flows of a configuration. Note that when considering configurations of the form $(\overset{\mathsf{oo}}{\rightsquigarrow}, \overset{\mathsf{os}}{\rightsquigarrow})$, a flow between objects granted by $\overset{\mathsf{oo}}{\rightsquigarrow}$ may lead a subject s, which is not allowed (by $\overset{\mathsf{os}}{\rightsquigarrow}$) to know the information contained into an object o, to access to this information if another subject allowed to read o generates a flow from o to an object o' such that s is allowed to know o' according to $\overset{\mathsf{os}}{\rightsquigarrow}$. Hence, it may be relevant to constrain configurations by considering the set $\mathcal{C}^{C}_{\mathsf{oos}} \subseteq \mathcal{C}_{\mathsf{oos}}$. Similarly, when considering configurations of the form $(\overset{\mathsf{oo}}{\rightsquigarrow}, \overset{\mathsf{so}}{\rightsquigarrow})$, a flow between objects granted by $\overset{\mathsf{oo}}{\rightsquigarrow}$ may lead a subject s, which is not allowed (by $\overset{\mathsf{so}}{\rightsquigarrow}$) to write information into an object o, to perform this action by writing into an object o' such that there is a granted flow from o' to o. We define $\mathcal{C}^{I}_{\mathsf{soo}} \subseteq \mathcal{C}_{\mathsf{soo}}$ in order to avoid such situation. Since composition of flows between objects and subjects leads to flows between objects, three subsets of $\mathcal{C}_{\mathsf{ci}}$ are defined for similar reasons.

III. EMBEDDINGS OF SECURITY POLICIES

A. Interpretation of security configurations

We introduce here the notion of embeddings of policies. Intuitively, we say that a policy $\mathbb{P}_1$ can be embedded into a policy $\mathbb{P}_2$ (based on the same set of targets than $\mathbb{P}_1$) iff the control done by $\mathbb{P}_1$ can be done by $\mathbb{P}_2$. Hence, $\mathbb{P}_1$ can be embedded into $\mathbb{P}_2$, iff each configuration c_1 of $\mathbb{P}_1$ can be interpreted by a configuration c_2 of $\mathbb{P}_2$ which authorizes exactly the same targets. This leads us to introduce interpretations of configurations of $\mathbb{P}_1$ by configurations of $\mathbb{P}_2$ allowing to define embedding operators.

[1]Given two relations $R_1 \subseteq A \times B$ and $R_2 \subseteq B \times C$, $R_2 \circ R_1 \subseteq A \times C$ is the relation defined by $\forall a, c \in A \times C$, $(a, c) \in R_2 \circ R_1$ iff $\exists b \in B$, such that $(a, b) \in R_1$ and $(b, c) \in R_2$.

Definition 2 (Embeddings): Let $\mathbb{P}_1 = (\mathbb{T}, \mathcal{C}_1, \Vdash_1)$ and $\mathbb{P}_2 = (\mathbb{T}, \mathcal{C}_2, \Vdash_2)$ be two security policies.

- $\mathbb{P}_1$ can be embedded into $\mathbb{P}_2$, which is written $\mathbb{P}_1 \trianglelefteq \mathbb{P}_2$, iff:
$$\forall c_1 \in \mathcal{C}_1\ \exists c_2 \in \mathcal{C}_2\ \forall t \in \mathbb{T} \quad c_1 \Vdash_1 t \Leftrightarrow c_2 \Vdash_2 t$$
- An interpretation I of configurations of $\mathbb{P}_1$ by configurations of $\mathbb{P}_2$ is a function $I : \mathcal{C}_1 \to \mathcal{C}_2$.
- An embedding operator of $\mathbb{P}_1$ into $\mathbb{P}_2$ is an interpretation $I : \mathcal{C}_1 \to \mathcal{C}_2$ such that:
$$\forall c_1 \in \mathcal{C}_1\ \forall t \in \mathbb{T} \quad c_1 \Vdash_1 t \Leftrightarrow I(c_1) \Vdash_2 t$$

In other words, $\mathbb{P}_1$ can be embedded into $\mathbb{P}_2$ iff for all configuration $c_1 \in \mathcal{C}_1$, there exists a configuration $c_2 \in \mathcal{C}_2$ such that $[\![c_1]\!]_{\mathbb{P}_1} = [\![c_2]\!]_{\mathbb{P}_2}$ and thus $c_2 = I(c_1)$ means that the configurations c_1 and c_2 allow exactly the same targets.

Hence, $\mathbb{P}_1 \trianglelefteq \mathbb{P}_2$ means that $\mathbb{P}_2$ has at least the same expressive power than $\mathbb{P}_1$. Of course, if there exists an embedding operator $I : \mathcal{C}_1 \to \mathcal{C}_2$, then $\mathbb{P}_1 \trianglelefteq \mathbb{P}_2$. However, note that such an embedding operator is not necessarily unique. Indeed, this is the case when $\mathbb{P}_2$ is defined from a set of configurations such that there exists several configurations associated with the same set of secure targets. The relation $\trianglelefteq$ clearly defines a preorder relation and we write $\mathbb{P}_1 \equiv \mathbb{P}_2$ when $\mathbb{P}_1 \trianglelefteq \mathbb{P}_2$ and $\mathbb{P}_2 \trianglelefteq \mathbb{P}_1$.

Example 1: If we consider the flow policies $\mathbb{P}_{\mathsf{oo}}$ and $\mathbb{P}^{T}_{\mathsf{oo}}$, introduced in table II, we have $\mathbb{P}^{T}_{\mathsf{oo}} \trianglelefteq \mathbb{P}_{\mathsf{oo}}$, since, clearly, the interpretation such that:
$$\forall \overset{\mathsf{oo}}{\rightsquigarrow} \in \mathcal{C}_{\mathsf{oo}} \quad I(\overset{\mathsf{oo}}{\rightsquigarrow}) = (\overset{\mathsf{oo}}{\rightsquigarrow})^{\star}$$
defines an embedding operator of $\mathbb{P}^{T}_{\mathsf{oo}}$ into $\mathbb{P}_{\mathsf{oo}}$.

B. Application to access control and flows policies

The relation $\trianglelefteq$ has been used to compare HRU and RBAC policies and also to compare the abstract flows policies introduced in table II. Of course, not surprisingly, HRU and RBAC policies have the same expressive power in terms of authorizations (which does not mean that they are equivalent from an administrative point of view). More formally, we have proved the following proposition.

Proposition 1:

1) For each pair of flow policies $(\mathbb{P}_1, \mathbb{P}_2)$ defined in table II, there is a path from $\mathbb{P}_1$ to $\mathbb{P}_2$ in figure 1 iff $\mathbb{P}_1 \trianglelefteq \mathbb{P}_2$.
2) $\mathbb{P}_{\mathsf{hru}} \equiv \mathbb{P}_{\mathsf{rbac}}$

TABLE II
ABSTRACT FLOW POLICIES

Policy	Configurations	$c \Vdash (\overset{\mathsf{oo}}{\curvearrowright}, \overset{\mathsf{os}}{\curvearrowright}, \overset{\mathsf{so}}{\curvearrowright}) \Leftrightarrow$
$\mathbb{P}_{\mathsf{oo}}$	$\mathcal{C}_{\mathsf{oo}} = \{\overset{\mathsf{oo}}{\rightsquigarrow} \subseteq \overset{\mathsf{oo}}{\hookrightarrow}\}$	$\overset{\mathsf{oo}}{\curvearrowright} \subseteq \overset{\mathsf{oo}}{\rightsquigarrow}$
$\mathbb{P}^T_{\mathsf{oo}}$	$\mathcal{C}_{\mathsf{oo}} = \{\overset{\mathsf{oo}}{\rightsquigarrow} \subseteq \overset{\mathsf{oo}}{\hookrightarrow}\}$	$\overset{\mathsf{oo}}{\curvearrowright} \subseteq (\overset{\mathsf{oo}}{\rightsquigarrow})^\star$
$\mathbb{P}_{\mathsf{os}}$	$\mathcal{C}_{\mathsf{os}} = \{\overset{\mathsf{os}}{\rightsquigarrow} \subseteq \overset{\mathsf{os}}{\hookrightarrow}\}$	$\overset{\mathsf{os}}{\curvearrowright} \subseteq \overset{\mathsf{os}}{\rightsquigarrow}$
$\mathbb{P}_{\mathsf{so}}$	$\mathcal{C}_{\mathsf{so}} = \{\overset{\mathsf{so}}{\rightsquigarrow} \mid \overset{\mathsf{so}}{\rightsquigarrow} \subseteq \overset{\mathsf{so}}{\hookrightarrow}\}$	$\overset{\mathsf{so}}{\curvearrowright} \subseteq \overset{\mathsf{so}}{\rightsquigarrow}$
$\mathbb{P}_{\mathsf{oos}}$	$\mathcal{C}_{\mathsf{oos}} = \{(\overset{\mathsf{oo}}{\rightsquigarrow}, \overset{\mathsf{os}}{\rightsquigarrow}) \subseteq \overset{\mathsf{oo}}{\hookrightarrow} \times \overset{\mathsf{os}}{\hookrightarrow}\}$	$\overset{\mathsf{oo}}{\curvearrowright} \subseteq \overset{\mathsf{oo}}{\rightsquigarrow} \wedge \overset{\mathsf{os}}{\curvearrowright} \subseteq \overset{\mathsf{os}}{\rightsquigarrow}$
$\mathbb{P}^T_{\mathsf{oos}}$	$\mathcal{C}_{\mathsf{oos}} = \{(\overset{\mathsf{oo}}{\rightsquigarrow}, \overset{\mathsf{os}}{\rightsquigarrow}) \subseteq \overset{\mathsf{oo}}{\hookrightarrow} \times \overset{\mathsf{os}}{\hookrightarrow}\}$	$\overset{\mathsf{oo}}{\curvearrowright} \subseteq (\overset{\mathsf{oo}}{\rightsquigarrow})^\star \wedge \overset{\mathsf{os}}{\curvearrowright} \subseteq \overset{\mathsf{os}}{\rightsquigarrow}$
$\mathbb{P}^C_{\mathsf{oos}}$	$\mathcal{C}^C_{\mathsf{oos}} = \left\{ (\overset{\mathsf{oo}}{\rightsquigarrow}, \overset{\mathsf{os}}{\rightsquigarrow}) \mid \forall o_1, o_2 \in \mathcal{O}\ \forall s \in \mathcal{S}\ (o_1 (\overset{\mathsf{oo}}{\rightsquigarrow})^\star o_2 \wedge o_2 \overset{\mathsf{os}}{\rightsquigarrow} s) \Rightarrow o_1 \overset{\mathsf{os}}{\rightsquigarrow} s \right\}$	$\overset{\mathsf{oo}}{\curvearrowright} \subseteq (\overset{\mathsf{oo}}{\rightsquigarrow})^\star \wedge \overset{\mathsf{os}}{\curvearrowright} \subseteq \overset{\mathsf{os}}{\rightsquigarrow}$
$\mathbb{P}_{\mathsf{soo}}$	$\mathcal{C}_{\mathsf{soo}} = \{(\overset{\mathsf{oo}}{\rightsquigarrow}, \overset{\mathsf{so}}{\rightsquigarrow}) \subseteq \overset{\mathsf{oo}}{\hookrightarrow} \times \overset{\mathsf{so}}{\hookrightarrow}\}$	$\overset{\mathsf{oo}}{\curvearrowright} \subseteq \overset{\mathsf{oo}}{\rightsquigarrow} \wedge \overset{\mathsf{so}}{\curvearrowright} \subseteq \overset{\mathsf{so}}{\rightsquigarrow}$
$\mathbb{P}^T_{\mathsf{soo}}$	$\mathcal{C}_{\mathsf{soo}} = \{(\overset{\mathsf{oo}}{\rightsquigarrow}, \overset{\mathsf{so}}{\rightsquigarrow}) \subseteq \overset{\mathsf{oo}}{\hookrightarrow} \times \overset{\mathsf{so}}{\hookrightarrow}\}$	$\overset{\mathsf{oo}}{\curvearrowright} \subseteq (\overset{\mathsf{oo}}{\rightsquigarrow})^\star \wedge \overset{\mathsf{so}}{\curvearrowright} \subseteq \overset{\mathsf{so}}{\rightsquigarrow}$
$\mathbb{P}^I_{\mathsf{soo}}$	$\mathcal{C}^I_{\mathsf{soo}} = \left\{ (\overset{\mathsf{oo}}{\rightsquigarrow}, \overset{\mathsf{so}}{\rightsquigarrow}) \mid \forall o_1, o_2 \in \mathcal{O}\ \forall s \in \mathcal{S}\ (s \overset{\mathsf{os}}{\rightsquigarrow} o_1 \wedge o_1 (\overset{\mathsf{oo}}{\rightsquigarrow})^\star o_2) \Rightarrow s \overset{\mathsf{os}}{\rightsquigarrow} o_2 \right\}$	$\overset{\mathsf{oo}}{\curvearrowright} \subseteq (\overset{\mathsf{oo}}{\rightsquigarrow})^\star \wedge \overset{\mathsf{so}}{\curvearrowright} \subseteq \overset{\mathsf{so}}{\rightsquigarrow}$
$\mathbb{P}_{\mathsf{ci}}$	$\mathcal{C}_{\mathsf{ci}} = \{(\overset{\mathsf{os}}{\rightsquigarrow}, \overset{\mathsf{so}}{\rightsquigarrow}) \subseteq \overset{\mathsf{os}}{\hookrightarrow} \times \overset{\mathsf{so}}{\hookrightarrow}\}$	$\overset{\mathsf{os}}{\curvearrowright} \subseteq \overset{\mathsf{os}}{\rightsquigarrow} \wedge \overset{\mathsf{so}}{\curvearrowright} \subseteq \overset{\mathsf{so}}{\rightsquigarrow}$
$\mathbb{P}^C_{\mathsf{ci}}$	$\mathcal{C}^C_{\mathsf{ci}} = \left\{ (\overset{\mathsf{os}}{\rightsquigarrow}, \overset{\mathsf{so}}{\rightsquigarrow}) \mid \forall o_1, o_2 \in \mathcal{O}\ \forall s \in \mathcal{S}\ ((o_1, o_2) \in (\overset{\mathsf{so}}{\rightsquigarrow} \circ \overset{\mathsf{os}}{\rightsquigarrow}) \wedge o_2 \overset{\mathsf{os}}{\rightsquigarrow} s) \Rightarrow o_1 \overset{\mathsf{os}}{\rightsquigarrow} s \right\}$	$\overset{\mathsf{os}}{\curvearrowright} \subseteq \overset{\mathsf{os}}{\rightsquigarrow} \wedge \overset{\mathsf{so}}{\curvearrowright} \subseteq \overset{\mathsf{so}}{\rightsquigarrow}$
$\mathbb{P}^I_{\mathsf{ci}}$	$\mathcal{C}^I_{\mathsf{ci}} = \left\{ (\overset{\mathsf{os}}{\rightsquigarrow}, \overset{\mathsf{so}}{\rightsquigarrow}) \mid \forall o_1, o_2 \in \mathcal{O}\ \forall s \in \mathcal{S}\ ((o_1, o_2) \in (\overset{\mathsf{so}}{\rightsquigarrow} \circ \overset{\mathsf{os}}{\rightsquigarrow}) \wedge s \overset{\mathsf{so}}{\rightsquigarrow} o_1) \Rightarrow s \overset{\mathsf{so}}{\rightsquigarrow} o_2 \right\}$	$\overset{\mathsf{os}}{\curvearrowright} \subseteq \overset{\mathsf{os}}{\rightsquigarrow} \wedge \overset{\mathsf{so}}{\curvearrowright} \subseteq \overset{\mathsf{so}}{\rightsquigarrow}$
$\mathbb{P}^{CI}_{\mathsf{ci}}$	$\mathcal{C}^{CI}_{\mathsf{ci}} = \mathcal{C}^C_{\mathsf{ci}} \cap \mathcal{C}^I_{\mathsf{ci}}$	$\overset{\mathsf{os}}{\curvearrowright} \subseteq \overset{\mathsf{os}}{\rightsquigarrow} \wedge \overset{\mathsf{so}}{\curvearrowright} \subseteq \overset{\mathsf{so}}{\rightsquigarrow}$

$$\begin{tikzcd}
\mathbb{P}_{\mathsf{soo}} & & & \mathbb{P}_{\mathsf{ci}} & & & \mathbb{P}_{\mathsf{oos}} \\
& & \mathbb{P}^C_{\mathsf{ci}} \arrow[ur] & & \mathbb{P}^I_{\mathsf{ci}} \arrow[ul] & & \\
\mathbb{P}^T_{\mathsf{soo}} \arrow[uu] & & & \mathbb{P}^{CI}_{\mathsf{ci}} \arrow[ul] \arrow[ur] & & & \mathbb{P}^T_{\mathsf{oos}} \arrow[uu] \\
\mathbb{P}_{\mathsf{so}} \arrow[u] \arrow[uurr] & & & \mathbb{P}_{\mathsf{oo}} \arrow[uuulll] \arrow[uuurrr] & & & \mathbb{P}_{\mathsf{os}} \arrow[u] \arrow[uull] \\
& & \mathbb{P}^I_{\mathsf{soo}} \arrow[uull] & & \mathbb{P}^C_{\mathsf{oos}} \arrow[uurr] & & \\
& & & \mathbb{P}^T_{\mathsf{oo}} \arrow[ul] \arrow[uu] \arrow[ur] & & &
\end{tikzcd}$$

Fig. 1. Hierarchy of abstract flow policies

IV. INTERPRETATION OF SECURITY POLICIES

A. Interpretation of security targets

Comparing two security policies according to the pre-order $\trianglelefteq$ requires that they share the same set of targets. In practice, this is not always the case. For example, characterizing flow properties induced by an access control policy first requires to interpret the access control policy by a flow policy and then to compare this flow policy with other known flow policies by using embeddings. Interpreting a security policy over a different set of targets leads to interpret its targets. In fact, we can generalize the semantics $[\![\,]\!]_{\mathbb{P}}$ of configurations of a policy $\mathbb{P}$ by considering interpretations of the set $\mathbb{T}$ of targets over an arbitrary domain $\mathbb{D}$.

Definition 3 (Interpreted policies): Let $\mathbb{P} = (\mathbb{T}, \mathcal{C}, \Vdash)$ be a policy. A $\mathbb{D}$-interpretation $\mathcal{I}$ of $\mathbb{T}$ is a mapping $[\![\,]\!]_{\mathcal{I}} : \mathbb{T} \to \mathbb{D}$ from which we define the interpreted policy:

$$[\![\mathbb{P}]\!]_{\mathcal{I}} = ([\![\mathbb{T}]\!]_{\mathcal{I}}, [\![\mathcal{C}]\!]_{\mathcal{I}}, [\![\Vdash]\!]_{\mathcal{I}})$$

where:

- $\forall T \subseteq \mathbb{T}\ \ [\![T]\!]_{\mathcal{I}} = \{[\![t]\!]_{\mathcal{I}} \mid t \in T\}$
- $\forall C \subseteq \mathcal{C}\ \ [\![C]\!]_{\mathcal{I}} = \{[\![c]\!]_{\mathcal{I}} \mid c \in C\}$
  where $\forall c \in \mathcal{C}\ \ [\![c]\!]_{\mathcal{I}} = \{[\![t]\!]_{\mathcal{I}} \mid c \Vdash t\} = [\![[\![c]\!]_{\mathbb{P}}]\!]_{\mathcal{I}}$
- $\forall c_{\mathcal{I}} \in [\![\mathcal{C}]\!]_{\mathcal{I}}\ \ \forall t_{\mathcal{I}} \in [\![\mathbb{T}]\!]_{\mathcal{I}} \quad c_{\mathcal{I}}\, [\![\Vdash]\!]_{\mathcal{I}}\, t_{\mathcal{I}} \Leftrightarrow t_{\mathcal{I}} \in c_{\mathcal{I}}$

In fact, a $\mathbb{D}$-interpretation $\mathcal{I}$ of targets of $\mathbb{P}$ provides a way to obtain an "extensional" representation $[\![\mathbb{P}]\!]_{\mathcal{I}}$ of $\mathbb{P}$, for which the targets are some elements of $\mathbb{D}$ and the configurations are some subsets of $\mathbb{D}$ (a configuration c is represented by the set of the interpretation of targets that c authorizes). Such a mechanism allows to give an uniform representation of several policies, and eases some analysis (such as comparison) or operations over policies (such as composition). Of course, interpretations of policies preserve embeddings as shown by the following proposition which is an immediate consequence of definitions.

Proposition 2: Let $\mathbb{P}_1 = (\mathbb{T}, \mathcal{C}_1, \Vdash_1)$ and $\mathbb{P}_2 = (\mathbb{T}, \mathcal{C}_2, \Vdash_2)$ be two security policies, and $\mathcal{I}$ be an interpretation of $\mathbb{T}$.

$$\mathbb{P}_1 \trianglelefteq \mathbb{P}_2 \Leftrightarrow [\![\mathbb{P}_1]\!]_{\mathcal{I}} \trianglelefteq [\![\mathbb{P}_2]\!]_{\mathcal{I}}$$

Remark 1: The semantics of configurations $[\![\,]\!]_{\mathbb{P}}$ can be defined by considering the identity $\mathbb{T}$-interpretation $Id : \mathbb{T} \to \mathbb{T}$ of $\mathbb{T}$ (i.e. $\forall t \in \mathbb{T}\ \ [\![t]\!]_{Id} = t$). Indeed, for all $c \in \mathcal{C}$, we have $[\![c]\!]_{\mathbb{P}} = [\![c]\!]_{Id}$. Hence, $\mathbb{P}_1 = (\mathbb{T}, \mathcal{C}_1, \Vdash_1) \trianglelefteq \mathbb{P}_2 = (\mathbb{T}, \mathcal{C}_2, \Vdash_2)$ iff $[\![\mathcal{C}_1]\!]_{\mathbb{P}_1} \subseteq [\![\mathcal{C}_2]\!]_{\mathbb{P}_2}$, and thus, $\mathbb{P}_1 \equiv \mathbb{P}_2$ means that $[\![\mathbb{P}_1]\!]_{Id}$ and $[\![\mathbb{P}_2]\!]_{Id}$ define the same security policy.

B. Flow-based interpretation of sets of accesses

In the next section, we will interpret access control policies by flow policies in order to compare them. Hence, we introduce here a $\mathbb{T}_F$-interpretation $\mathcal{I}_F : \mathbb{T}_A \to \mathbb{T}_F$ of $\mathbb{T}_A$ allowing to characterize flows generated by a set of accesses.

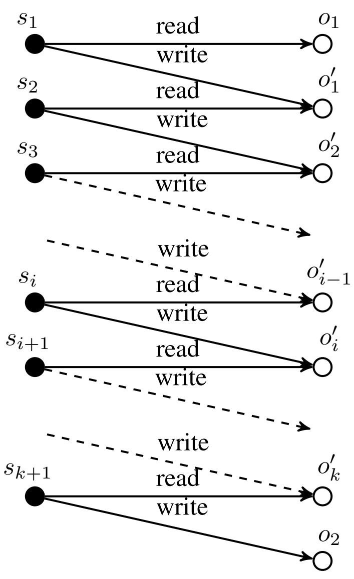

Fig. 2. Information flows

Indeed, accesses done in the system generate information flows between objects, and between subjects and objects. An elementary flow of the information contained into an object o_1 to an object o_2 can occur iff there exists a subject s reading o_1 and writing into o_2.

Definition 4: Given a set $E \in \mathbb{T}_A$ of accesses, the set of elementary flows generated by E is defined by:

$$\mapsto_E = \left\{ o_1 \overset{\text{oo}}{\hookrightarrow} o_2 \mid \exists s \in \mathcal{S}\ \{(s, o_1, \text{read}), (s, o_2, \text{write})\} \subseteq E \right\}$$

We can now introduce a $\overset{\text{oo}}{\hookrightarrow}$-interpretation $\mathcal{I}_{\text{oo}}$ of $\mathbb{T}_A$ allowing to characterize flows between objects generated by a set E of accesses: the information contained into o_1 can flow into o_2 (in this case, we write $o_1 \overset{\text{oo}}{\hookrightarrow}_E o_2$ instead of $(o_1, o_2) \in [\![E]\!]_{\mathcal{I}_{\text{oo}}}$) iff there exists in E a chain of read and write accesses starting from a read access over o_1 and ending at a write access over o_2. The flows between objects generated by a set E of accesses are thus simply defined as the reflexive and transitive closure of $\mapsto_E$.

Definition 5: $[\![E]\!]_{\mathcal{I}_{\text{oo}}} = \overset{\text{oo}}{\hookrightarrow}_E = \mapsto_E^{\star}$

Hence (see figure 2), we have $o_1 \overset{\text{oo}}{\hookrightarrow}_E o_2$ iff either $o_1 = o_2$ or:

$$\exists s_1, \cdots, s_k, s_{k+1} \in \mathcal{S}\ \exists o'_1, \cdots, o'_k \in \mathcal{O}$$
$$\left\{ \begin{array}{l} (s_1, o_1, \text{read}), (s_1, o'_1, \text{write}), \\ (s_2, o'_1, \text{read}), (s_2, o'_2, \text{write}), \cdots, \\ (s_i, o'_{i-1}, \text{read}), (s_i, o'_i, \text{write}), \cdots, \\ (s_{k+1}, o'_k, \text{read}), (s_{k+1}, o_2, \text{write}) \end{array} \right\} \subseteq E$$

Example 2: The set of flows generated by the set of accesses:

$$E = \left\{ \begin{array}{lll} (s_1, o_1, \text{read}), & (s_1, o_1, \text{write}), & (s_1, o_3, \text{read}), \\ (s_2, o_1, \text{read}), & (s_2, o_2, \text{read}), & (s_2, o_2, \text{write}), \\ (s_3, o_2, \text{read}), & (s_3, o_2, \text{write}), & (s_3, o_4, \text{write}) \end{array} \right\}$$

is defined by :

$$\overset{\text{oo}}{\hookrightarrow}_E = \left\{ \begin{array}{llll} o_1 \overset{\text{oo}}{\hookrightarrow} o_1, & o_2 \overset{\text{oo}}{\hookrightarrow} o_2, & o_3 \overset{\text{oo}}{\hookrightarrow} o_3, & o_4 \overset{\text{oo}}{\hookrightarrow} o_4, \\ o_3 \overset{\text{oo}}{\hookrightarrow} o_1, & o_1 \overset{\text{oo}}{\hookrightarrow} o_2, & o_2 \overset{\text{oo}}{\hookrightarrow} o_4, & o_1 \overset{\text{oo}}{\hookrightarrow} o_4, \\ o_3 \overset{\text{oo}}{\hookrightarrow} o_2, & o_3 \overset{\text{oo}}{\hookrightarrow} o_4 & & \end{array} \right\}$$

Remark 2: Note that we assume here the "worst" case: we suppose here that if there is a potential for information flow then the flow actually occurs. Indeed, the definition of $\overset{\text{oo}}{\hookrightarrow}_E$ over-estimates the information flow generated by a set of accesses. In fact, this definition does not take into account the temporal aspect of accesses since a set of accesses is unordered. However, it is possible to refine this definition by considering sequences of flows corresponding to ordered subsets of the set $\mapsto_E$ of elementary flows generated by a set E of accesses. For example, this can be done by observing flows at the operating system level. More formally, from a set of accesses E, if the behavior of the system can be described by a sequence S of elementary flows, we can define flows occurring during this sequence as follows:

$$\overset{\text{oo}}{\rightarrowtail}_S = \left\{ \begin{array}{l} \emptyset \text{ if } S = (\,) \\ \{o_1 \overset{\text{oo}}{\hookrightarrow} o_2\} \cup \overset{\text{oo}}{\rightarrowtail}_{S'} \cup \{o \overset{\text{oo}}{\hookrightarrow} o_2 \mid o \overset{\text{oo}}{\rightarrowtail}_{S'} o_1\} \\ \quad \text{if } S = S'.(o_1 \overset{\text{oo}}{\hookrightarrow} o_2) \end{array} \right.$$

Of course, elementary flows occurring in S have to be realizable according to E, and we have $\overset{\text{oo}}{\rightarrowtail}_S \subseteq \overset{\text{oo}}{\hookrightarrow}_E$.

From the definition of $\overset{\text{oo}}{\hookrightarrow}_E$, we can now introduce the $\overset{\text{os}}{\hookrightarrow}$-interpretation $\mathcal{I}_{\text{os}}$ and the $\overset{\text{so}}{\hookrightarrow}$-interpretation $\mathcal{I}_{\text{so}}$ of $\mathbb{T}_A$ allowing to characterize flows occurring between subjects and objects which are generated by a set of accesses E:

$$[\![E]\!]_{\mathcal{I}_{\text{os}}} = \overset{\text{os}}{\hookrightarrow}_E = \left\{ o_1 \overset{\text{os}}{\hookrightarrow} s \mid o_1 \overset{\text{oo}}{\hookrightarrow}_E o_2 \wedge (s, o_2, \text{read}) \in E \right\}$$
$$[\![E]\!]_{\mathcal{I}_{\text{so}}} = \overset{\text{so}}{\hookrightarrow}_E = \left\{ s \overset{\text{so}}{\hookrightarrow} o_2 \mid (s, o_1, \text{write}) \in E \wedge o_1 \overset{\text{oo}}{\hookrightarrow}_E o_2 \right\}$$

Example 3: If we consider again the set E of accesses introduced in example 2, we have:

$$\overset{\text{os}}{\hookrightarrow}_E = \left\{ \begin{array}{llll} o_1 \overset{\text{os}}{\hookrightarrow} s_1, & o_3 \overset{\text{os}}{\hookrightarrow} s_1, & o_1 \overset{\text{os}}{\hookrightarrow} s_2, & o_2 \overset{\text{os}}{\hookrightarrow} s_2, \\ o_3 \overset{\text{os}}{\hookrightarrow} s_2, & o_1 \overset{\text{os}}{\hookrightarrow} s_3, & o_2 \overset{\text{os}}{\hookrightarrow} s_3, & o_3 \overset{\text{os}}{\hookrightarrow} s_3 \end{array} \right\}$$

$$\overset{\text{so}}{\hookrightarrow}_E = \left\{ \begin{array}{llll} s_1 \overset{\text{so}}{\hookrightarrow} o_1, & s_1 \overset{\text{so}}{\hookrightarrow} o_2, & s_2 \overset{\text{so}}{\hookrightarrow} o_2, & s_1 \overset{\text{so}}{\hookrightarrow} o_4, \\ s_2 \overset{\text{so}}{\hookrightarrow} o_4, & s_3 \overset{\text{so}}{\hookrightarrow} o_4 & & \end{array} \right\}$$

Of course, flows between objects are generated by flows from subjects to objects and by flows from objects to subjects, and we have proved the following proposition.

Proposition 3: $\forall E \in \mathbb{T}_A \quad (\overset{\text{oo}}{\hookrightarrow}_E, \overset{\text{os}}{\hookrightarrow}_E, \overset{\text{so}}{\hookrightarrow}_E) \in \mathbb{T}_F$

Remark 3: Conversely, we have proved that for any target $(\overset{\text{oo}}{\curvearrowright}, \overset{\text{os}}{\curvearrowright}, \overset{\text{so}}{\curvearrowright}) \in \mathbb{T}_F$, the set of accesses:

$$E = \left\{ (s, o, \text{read}) \mid o \overset{\text{os}}{\curvearrowright} s \right\} \cup \left\{ (s, o, \text{write}) \mid s \overset{\text{so}}{\curvearrowright} o \right\} \in \mathbb{T}_A$$

is such that $(\overset{\text{oo}}{\hookrightarrow}_E, \overset{\text{os}}{\hookrightarrow}_E, \overset{\text{so}}{\hookrightarrow}_E) = (\overset{\text{oo}}{\curvearrowright}, \overset{\text{os}}{\curvearrowright}, \overset{\text{so}}{\curvearrowright})$.

Thanks to proposition 3, we finally introduce the flow-based interpretation of access control policies defined as the $\mathbb{T}_F$-interpretation $\mathcal{I}_F$ of $\mathbb{T}_A$ such that:

$$[\![E]\!]_{\mathcal{I}_F} = ([\![E]\!]_{\mathcal{I}_{\text{oo}}}, [\![E]\!]_{\mathcal{I}_{\text{os}}}, [\![E]\!]_{\mathcal{I}_{\text{so}}}) = (\overset{\text{oo}}{\hookrightarrow}_E, \overset{\text{os}}{\hookrightarrow}_E, \overset{\text{so}}{\hookrightarrow}_E)$$

V. Equivalent security policies

A. Proving equivalence between policies

Interpretations of configurations and interpretations of targets are both semantical mechanisms allowing to build bridges between security policies and they can be combined to compare arbitrary security policies. In practice, proving that interpreting a policy $\mathbb{P}_1$ with an interpretation (of targets) $\mathcal{I}$ leads to a policy equivalent (by embeddings) to a policy $\mathbb{P}_2$, can be done by mapping each configuration $c_1 \in \mathcal{C}_1$ by an "equivalent" configuration $c_2 \in \mathcal{C}_2$. The following proposition formally explicit such a construction.

Proposition 4: Let $\mathbb{P}_1 = (\mathbb{T}_1, \mathcal{C}_1, \Vdash_1)$, $\mathbb{P}_2 = (\mathbb{T}_2, \mathcal{C}_2, \Vdash_2)$ be two security policies, and $\mathcal{I} : \mathbb{T}_1 \to \mathbb{T}_2$ be a $\mathbb{T}_2$-interpretation of $\mathbb{T}_1$. $\llbracket \mathbb{P}_1 \rrbracket_{\mathcal{I}} \equiv \mathbb{P}_2$ iff there exists an interpretation $I : \mathcal{C}_1 \to \mathcal{C}_2$ such that:

$$\llbracket \mathbb{P}_1 \rrbracket_{\mathcal{I}} = \llbracket (\mathbb{T}_2, I(\mathcal{C}_1), \Vdash_2) \rrbracket_{Id} = \llbracket \mathbb{P}_2 \rrbracket_{Id}$$

Proof: ($\Rightarrow$) We write $\mathbb{P}_2'$ the policy $(\mathbb{T}_2, I(\mathcal{C}_1), \Vdash_2)$. First note that since $\llbracket \mathbb{P}_1 \rrbracket_{\mathcal{I}} \equiv \mathbb{P}_2$, we have $\mathbb{T}_2 = \llbracket \mathbb{T}_1 \rrbracket_{\mathcal{I}}$ (hence, $\mathcal{I}$ is surjective). Furthermore, for all $T_2 \subseteq \mathbb{T}_2$ such that $T_2 \in \llbracket \mathcal{C}_1 \rrbracket_{\mathcal{I}}$ (where $T_2 = \llbracket c_1 \rrbracket_{\mathcal{I}}$ for a configuration $c_1 \in \mathcal{C}_1$), by hypothesis, there exists $c_2 \in \mathcal{C}_2$ such that for all $t_2 \in \mathbb{T}_2$, $t_2 \in T_2$ iff $c_2 \Vdash_2 t_2$. Hence, there exists an interpretation $I : \mathcal{C}_1 \to \mathcal{C}_2$ such that for all $c_1 \in \mathcal{C}_1$, for all $t_2 \in \mathbb{T}_2$, $t_2 \in \llbracket c_1 \rrbracket_{\mathcal{I}}$ iff $I(c_1) \Vdash_2 t_2$. We first prove that $\llbracket \mathbb{P}_1 \rrbracket_{\mathcal{I}} = \llbracket \mathbb{P}_2' \rrbracket_{Id}$. Equality of sets of targets follows from definitions ($\llbracket \mathbb{T}_1 \rrbracket_{\mathcal{I}} = \mathbb{T}_2 = \llbracket \mathbb{T}_2 \rrbracket_{Id}$). Equality of sets of configurations is also obtained by definition as follows:

$$\begin{aligned}
\llbracket \mathcal{C}_1 \rrbracket_{\mathcal{I}} &= \{ \llbracket c_1 \rrbracket_{\mathcal{I}} \mid c_1 \in \mathcal{C}_1 \} \\
&= \{ \{ \llbracket t_1 \rrbracket_{\mathcal{I}} \mid c_1 \Vdash_1 t_1 \} \mid c_1 \in \mathcal{C}_1 \} \\
&= \{ \{ \llbracket t_1 \rrbracket_{\mathcal{I}} \mid \llbracket t_1 \rrbracket_{\mathcal{I}} \in \llbracket c_1 \rrbracket_{\mathcal{I}} \} \mid c_1 \in \mathcal{C}_1 \} \\
&= \{ \{ \llbracket t_1 \rrbracket_{\mathcal{I}} \mid I(c_1) \Vdash_2 \llbracket t_1 \rrbracket_{\mathcal{I}} \} \mid c_1 \in \mathcal{C}_1 \} \\
&= \left\{ \llbracket I(c_1) \rrbracket_{\mathbb{P}_2'} \mid c_1 \in \mathcal{C}_1 \right\} \\
&= \llbracket I(\mathcal{C}_1) \rrbracket_{\mathbb{P}_2'} = \llbracket I(\mathcal{C}_1) \rrbracket_{Id} \text{ (for } \mathbb{P}_2')
\end{aligned} \tag{1}$$

Last it remains to prove that $\llbracket \Vdash_1 \rrbracket_{\mathcal{I}} = \llbracket \Vdash_2 \rrbracket_{Id}$ (for $\mathbb{P}_2'$). Indeed, for all $t_1 \in \mathbb{T}_1$ and $c_1 \in C_1$, we have:

$$\begin{aligned}
& \llbracket c_1 \rrbracket_{\mathcal{I}} \llbracket \Vdash_1 \rrbracket_{\mathcal{I}} \llbracket t_1 \rrbracket_{\mathcal{I}} \\
\Leftrightarrow\ & \llbracket t_1 \rrbracket_{\mathcal{I}} \in \llbracket c_1 \rrbracket_{\mathcal{I}} \\
\Leftrightarrow\ & \llbracket t_1 \rrbracket_{\mathcal{I}} \in \llbracket I(c_1) \rrbracket_{\mathbb{P}_2'} && \text{(by (1))} \\
\Leftrightarrow\ & \llbracket I(c_1) \rrbracket_{\mathbb{P}_2'} \llbracket \Vdash_2 \rrbracket_{Id} \llbracket t_1 \rrbracket_{\mathcal{I}} \\
\Leftrightarrow\ & \llbracket c_1 \rrbracket_{\mathcal{I}} \llbracket \Vdash_2 \rrbracket_{Id} \llbracket t_1 \rrbracket_{\mathcal{I}} && \text{(by (1))}
\end{aligned}$$

Now, to obtain $\llbracket \mathbb{P}_2' \rrbracket_{Id} = \llbracket \mathbb{P}_2 \rrbracket_{Id}$, it remains to prove $\llbracket I(\mathcal{C}_1) \rrbracket_{Id} = \llbracket \mathcal{C}_2 \rrbracket_{Id}$ which can be obtained from (1) since $\llbracket \llbracket \mathcal{C}_1 \rrbracket_{\mathcal{I}} \rrbracket_{Id} = \llbracket \mathcal{C}_1 \rrbracket_{\mathcal{I}} = \llbracket \mathcal{C}_2 \rrbracket_{Id}$ (because $\llbracket \mathbb{P}_1 \rrbracket_{\mathcal{I}} \equiv \mathbb{P}_2$).

($\Leftarrow$) By hypothesis, the interpretation $I_1 : \llbracket \mathcal{C}_1 \rrbracket_{\mathcal{I}} \to \mathcal{C}_2$ such that:

$$\forall c_1 \in \mathcal{C}_1 \quad I_1(\llbracket c_1 \rrbracket_{\mathcal{I}}) = I(c_1)$$

is clearly an embedding operator of $\llbracket \mathbb{P}_1 \rrbracket_{\mathcal{I}}$ into $\mathbb{P}_2$, thus proving $\llbracket \mathbb{P}_1 \rrbracket_{\mathcal{I}} \trianglelefteq \mathbb{P}_2$. Similarly, since, by hypothesis, $\llbracket I(\mathcal{C}_1) \rrbracket_{Id} = \llbracket \mathcal{C}_2 \rrbracket_{Id}$, for all $c_2 \in \mathcal{C}_2$ there exists at least one configuration $c_1 \in \mathcal{C}_1$ such that $I(c_1) = c_2$. Hence, by hypothesis, there exists an interpretation $I_2 : \mathcal{C}_2 \to \llbracket \mathcal{C}_1 \rrbracket_{\mathcal{I}}$ such that:

$$I_2(c_2) = \llbracket c_1 \rrbracket_{\mathcal{I}} \quad \text{where } I(c_1) = c_2$$

$$\begin{tikzcd}
\mathbb{P}_{\mathsf{soo}} & \llbracket \mathbb{P}_{\mathsf{hru}} \rrbracket_{\mathcal{I}_F} \arrow[rrrr, leftrightarrow, bend left] & & \mathbb{P}_{\mathsf{ci}} & & \llbracket \mathbb{P}_{\mathsf{rbac}} \rrbracket_{\mathcal{I}_F} & \mathbb{P}_{\mathsf{oos}} \\
& & \mathbb{P}_{\mathsf{ci}}^{C} \arrow[ur] & & \mathbb{P}_{\mathsf{ci}}^{I} \arrow[ul] & & \\
\mathbb{P}_{\mathsf{soo}}^{T} \arrow[uu] & & & & & & \mathbb{P}_{\mathsf{oos}}^{T} \arrow[uu] \\
& & & \mathbb{P}_{\mathsf{ci}}^{CI} \arrow[uuull, bend left] \arrow[uuurr, bend right] \arrow[uul] \arrow[uur] & & & \\
\mathbb{P}_{\mathsf{so}} \arrow[uu] \arrow[uuurr] & & & \mathbb{P}_{\mathsf{oo}} \arrow[uuuulll] \arrow[uuuurrr] & & & \mathbb{P}_{\mathsf{os}} \arrow[uu] \arrow[uuull] \\
\llbracket \mathbb{P}_{\mathsf{mls}}^{I} \rrbracket_{\mathcal{I}_F} \arrow[rr, leftrightarrow] & & \mathbb{P}_{\mathsf{soo}}^{I} \arrow[uuull] & & \mathbb{P}_{\mathsf{oos}}^{C} \arrow[uuurr] \arrow[rr, leftrightarrow] & & \llbracket \mathbb{P}_{\mathsf{mls}}^{C} \rrbracket_{\mathcal{I}_F} \\
\llbracket \mathbb{P}_{\mathsf{mls}}^{\downarrow} \rrbracket_{\mathcal{I}_F} \arrow[rrr, leftrightarrow] & & & \mathbb{P}_{\mathsf{oo}}^{T} \arrow[ul] \arrow[uu] \arrow[ur] \arrow[rrr, leftrightarrow] & & & \llbracket \mathbb{P}_{\mathsf{mls}}^{\uparrow} \rrbracket_{\mathcal{I}_F}
\end{tikzcd}$$

Fig. 3. Hierarchy of security policies

which is an embedding operator of $\mathbb{P}_2$ into $\llbracket \mathbb{P}_1 \rrbracket_{\mathcal{I}}$. thus proving $\mathbb{P}_2 \trianglelefteq \llbracket \mathbb{P}_1 \rrbracket_{\mathcal{I}}$. ■

B. From access control policies to flow properties

We are now in position to complete the hierarchy of flow policies by considering the flow-based interpreted access control policies introduced in table I. The complete hierarchy is represented by figure 3. Indeed, we have proved the following proposition.

Proposition 5: For each pair of flow policies $(\mathbb{P}_1, \mathbb{P}_2)$, there is a path from $\mathbb{P}_1$ to $\mathbb{P}_2$ in figure 3 iff $\mathbb{P}_1 \trianglelefteq \mathbb{P}_2$.

Remark 4: Only the flow-based interpretation of $\mathbb{P}_{\mathsf{hru}}$ has been considered. Results about the flow-based interpretation of $\mathbb{P}_{\mathsf{rbac}}$ have been obtained by propositions 1.2 and 2.

Remark 5: The "no-write-down" and "no-write-up" policies are equivalent up to an inversion of the partial order over security levels, and thus can both be viewed as a confinment policy.

Proposition 5 formally justifies why MLS policies can be interpreted as flow policies. Indeed, for each MLS access control policy $\mathbb{P}_A = (\mathbb{T}_A, \mathcal{C}_A, \Vdash_A)$, there exists a flow policy $\mathbb{P}_F = (\mathbb{T}_F, \mathcal{C}_F, \Vdash_F)$ (defined in table II) such that $\llbracket \mathbb{P}_A \rrbracket_{\mathcal{I}_F} \equiv \mathbb{P}_F$. This means that $\mathbb{P}_A$ and $\mathbb{P}_F$ have the same expressive power to control flows. In other words, flows generated by authorized sets of accesses specified by configurations in $\mathcal{C}_A$ and authorized flows specified by configurations in $\mathcal{C}_F$ are the same. This is formally expressed by the equality

$$\llbracket \mathcal{C}_A \rrbracket_{\mathcal{I}_F} = \{ \llbracket c \rrbracket_{\mathbb{P}_F} \mid c \in \mathcal{C}_F \}$$

which can be obtained from $\llbracket \mathbb{P}_A \rrbracket_{\mathcal{I}_F} \equiv \mathbb{P}_F$. More precisely, by proposition 4, there exists an interpretation $I_C : \mathcal{C}_A \to \mathcal{C}_F$ such that:

$$\llbracket \mathbb{P}_A \rrbracket_{\mathcal{I}_{\mathcal{F}}} = \llbracket (\mathbb{T}_F, I_C(\mathcal{C}_A), \Vdash_F) \rrbracket_{Id} = \llbracket \mathbb{P}_F \rrbracket_{Id}$$

This means that:

$$\begin{aligned}
&\forall c_A \in \mathcal{C}_A \;\; \llbracket c_A \rrbracket_{\mathcal{I}_F} = \llbracket I_C(c_A) \rrbracket_{\mathbb{P}_F} \\
&\forall c_F \in \mathcal{C}_F \; \exists c_A \in \mathcal{C}_A \; c_F = I_C(c_A) \wedge \llbracket c_F \rrbracket_{\mathbb{P}_F} = \llbracket c_A \rrbracket_{\mathcal{I}_F}
\end{aligned}$$

For example, the equivalence $\left[\!\left[\mathbb{P}^{\uparrow}_{\mathsf{mls}}\right]\!\right]_{\mathcal{I}_F} \equiv \mathbb{P}^{T}_{\mathsf{oo}}$ can be proved by considering the interpretation $I_C : \mathcal{C}_{\mathsf{mls}} \to \mathcal{C}_{\mathsf{oo}}$ defined by:

$$I_C((\mathcal{L}, \preceq, f_O)) = \{o_1 \overset{\mathsf{oo}}{\hookrightarrow} o_2 \mid f_O(o_1) \preceq f_O(o_2)\}$$

Such a definition can be generalized since it leads us to consider I_C as an embedding operator of the flow-based interpretation of $\mathbb{P}_A$ into $\mathbb{P}_F$ (interpreted by the identity $\mathbb{T}_F$-interpretation Id) and thus, given a configuration $c_A \in \mathcal{C}_A$, $I_C(c_A)$ specifies the following sets of authorized flows:

$$\begin{aligned}
\overset{\mathsf{oo}}{\rightsquigarrow}_{c_A} &= \{o_1 \overset{\mathsf{oo}}{\hookrightarrow} o_2 \mid \exists E \in \mathbb{T}_A \ c_A \Vdash_A E \wedge \ o_1 \overset{\mathsf{oo}}{\hookrightarrow}_E o_2\} \\
\overset{\mathsf{os}}{\rightsquigarrow}_{c_A} &= \{o \overset{\mathsf{os}}{\hookrightarrow} s \mid \exists E \in \mathbb{T}_A \ c_A \Vdash_A E \wedge \ (s, o, \text{read}) \in E\} \\
\overset{\mathsf{so}}{\rightsquigarrow}_{c_A} &= \{s \overset{\mathsf{os}}{\hookrightarrow} o \mid \exists E \in \mathbb{T}_A \ c_A \Vdash_A E \wedge \ (s, o, \text{write}) \in E\}
\end{aligned}$$

On the contrary, for access control policies which are not equivalent to a flow policy, like HRU or RBAC, there exists at least a configuration $c_A \in \mathcal{C}_A$ such that $[\![c_A]\!]_{\mathbb{P}_A}$ contains a set of accesses generating an illegal flow according to $\overset{\mathsf{oo}}{\rightsquigarrow}_{c_A}$, $\overset{\mathsf{os}}{\rightsquigarrow}_{c_A}$, or $\overset{\mathsf{so}}{\rightsquigarrow}_{c_A}$ and in this case we have:

$$[\![c_A]\!]_{\mathcal{I}_F} \not\subseteq [\![I_C(c_A)]\!]_{\mathbb{P}_F}$$

For example, if we consider the configuration c_A of HRU specifying that authorized accesses are those in the set E defined in example 2:

- s_1 can read o_3 and write into o_1 on which s_2 can make a read access, even if s_2 cannot read o_3 (hence the flow $o_3 \overset{\mathsf{os}}{\hookrightarrow} s_2$ can be generated by an authorized set of accesses according to c_A while it is not authorized by $\overset{\mathsf{os}}{\rightsquigarrow}_{c_A}$ since a target containing the access (s_2, o_3, read) is not secure according to c_A)
- s_2 can write into o_2 and then s_3 can read o_2 and write into o_4, even if s_2 cannot write into o_4 (hence the flow $s_2 \overset{\mathsf{so}}{\hookrightarrow} o_4$ can be generated by an authorized set of accesses according to c_A while it is not authorized by $\overset{\mathsf{so}}{\rightsquigarrow}_{c_A}$ since a target containing the access (s_2, o_4, write) is not secure according to c_A)

In fact, only a subset of configurations of such policies can be viewed as a "coherent" (according to $\mathcal{C}^{CI}_{\mathsf{ci}}$) specification of authorized flows. However, as we said, enforcing the HRU policy with a configuration $c_A \in \mathcal{C}_{\mathsf{hru}}$ is clearly not sufficient to control flows according to $\overset{\mathsf{oo}}{\rightsquigarrow}_{c_A}$, $\overset{\mathsf{os}}{\rightsquigarrow}_{c_A}$, and $\overset{\mathsf{so}}{\rightsquigarrow}_{c_A}$. Indeed, if we consider again the configuration $c_A = E$ where E is defined in example 2, we would like for example that:

- s_2 can read o_1 while o_1 does not contain information coming from objects that s_2 is not authorized to read
- s_3 can write into o_4 while objects read by s_3 do not contain information coming from subjects that are not authorized to write into o_4

Of course, such a control cannot be done by only observing sets of accesses since it requires to know the origin of information contained into objects. Hence, in this case, a supplementary information has to be considered together with sets of accesses. For example, in [11], [12], [13], [14], intrusion detection systems have been defined by considering a tagging system over objects allowing to detect illegal information flows according to the flow policies $\overset{\mathsf{oo}}{\rightsquigarrow}_{c_A}$, $\overset{\mathsf{os}}{\rightsquigarrow}_{c_A}$, and $\overset{\mathsf{so}}{\rightsquigarrow}_{c_A}$ induced by the configuration c_A of an access control policy. In [3], such a tagging system (together with the proofs of its soundness and its completeness) has been formalised in our framework. Hence, this paper may also be viewed as the development *a posteriori* of theoretical foundations on which such mechanisms are based.

VI. Related work

Although there exist some papers focusing on comparisons and translations between security policies such as [15], [16], [17], these developments are not expressed within a common framework. Hence, it is rather difficult to compare (or to compose) these translations. In a previous work [18], we have introduced a comparison mechanism between policies sharing the same set of targets and based on the notion of simulation of transition systems enforcing the policies. We have compared the Bell & LaPadula, Chinese Wall and the RBAC policies according to this mechanism. Such an approach is similar to the one introduced in [19]. A more different approach is introduced in [20], where the comparison of the expressive power of access control models is based on security analysis. However, here again, the definition of reduction they introduced to compare access control models can be seen as a relaxation of a notion of simulation. In [15], the equivalence between a core RBAC model (without roles hierarchy and constraints) and the ACL (Access Control Lists) model is presented. Moreover, a way to extend this equivalence to an RBAC model with a roles hierarchy (as the one we consider) is stated. However, their comparison is not done by first introducing a generic framework to express access control models and then using it in order to compare the two models. Thus, we cannot really put into perspective the work of the author and ours if not by observing that his conclusion that RBAC is equivalent to ACL matches ours that RBAC is equivalent to HRU. Similarly, in [16], the authors concentrate on the fact that RBAC can be used to simulate Lattice Based Access Control (LBAC) and Discretionnary Access Control (DAC), by showing systematic constructions of RBAC in order to enforce those two kinds of access control. Here again, this work is not developed in a common framework, so this limits our interest in it, and the investigation of equivalences or absence of equivalences between RBAC and LBAC on the one hand, and, RBAC and DAC on the other hand has not be done, whereas we have shown that RBAC is not equivalent to MLS policies but that it is equivalent to HRU.

VII. Conclusion – Future work

The goal of security policies is to achieve security properties. On the other hand, the relation between these properties and the properties on which security mechanisms are based is not always obvious. Hence, a question naturally emerges: how to bridge the gap in such a case? In this paper, to answer this question, we have introduced formal definitions based on the semantics of security policies providing methodological tools allowing to handle such situations. More precisely, this paper addresses the problem of comparing security policies, focusing in particular on the relationship between access control policies on one side and flow control policies on the other. Our approach has been described within a generic framework previously introduced in [1] and allowing to define security policies. In fact, when designing this framework, our main motivation was to be able to provide some methodological guidelines to specify security policies but also to be able to compare these policies. Defining policies within a common formalism allows to ease their comparison. This framework can be used to consider many developments since it does not depend on the syntax used to described policies but only on some semantics aspects of policies. It allows to identify the various entities involved in the definition of a policy and their role. The main developments done within our framework is concerned with flow analysis of access control policies. In fact, access control policies allow to grant or to revoke the rights for some subjects to access some objects, but cannot always control how the information is used once it has been accessed (there is no control on its propagation). Intuitively, a link is needed between "what you do" (the policy) and "what you want" (the goal for which the policy is designed). In this paper, we have formalized this link through notions of interpretations. Of course results about access control and flow control expressed in proposition 5 may seem unsurprising. However, we think they allow us to gain a better understanding of the flow control induced by an access control and to define additional mechanisms to ensure flow properties (i.e. mechanisms allowing to detect illegal information flows). Moreover, our approach also provides a way to reuse the same specification of a security property in order to analyse or to verify several policies and systems, thus showing the benefits of a library of generic security properties, dedicated to particular domains (like information flows) and that can be considered in several contexts. For example, it becomes possible to check the same abstract information flow property expressed by a specification for several access control policies.

Comparing and understanding relationship between policies requires attention in emerging contexts where policies under different domains might need to be compared or merged. Our work can be viewed as a first step in this direction. However, several issues remain to be addressed. For example, we are currently working on administrative policies. Indeed, our framework only consider static policies in the sense that security configurations correspond to a static information. Nevertheless, in practice, security configurations are modified during the lifetime of a system and it seems desirable to control such transformations in order to preserve some security properties. This leads us to consider administrative policies whose targets are configurations of the administrated policies. We also plan to focus on composition of security policies, which can be very useful in a distributed context. We would like to define a language allowing to express how several policies can be combined and to characterize the security property induced by this combination (expressed from the properties of the initial policies).

Acknowledgment

The author would like to thank L. Habib, T. Hardin, L. Mé and V. Viet Triem Tong for enlightening discussions about access control policies flow policies, and their comparison, as well as the anonymous referees for their very useful comments.

References

[1] M. Jaume, "Security rules versus security properties," in *Information Systems Security - 6th International Conference, ICISS 2010, Proceedings*, ser. LNCS, vol. 6503. Springer, 2010, pp. 231–245.

[2] D. Doligez, M. Jaume, and R. Rioboo, "Development of secured systems by mixing programs, specifications and proofs in an object-oriented programming environment," in *ACM SIGPLAN Seventh Workshop on Programming Languages and Analysis for Security (PLAS 2012) Proceedings*. To appear, 2012.

[3] M. Jaume, V. Viet Triem Tong, and L. Mé, "Flow-based interpretation of access control: Detection of illegal information flows," in *Information Systems Security - 7th International Conference, ICISS 2011, Proceedings*, ser. LNCS, vol. 7093. Springer, 2011, pp. 72–86.

[4] T. Bourdier, H. Cirstea, M. Jaume, and H. Kirchner, "Formal specification and validation of security policies," in *Foundations & Practice of Security, FPS*, ser. LNCS, vol. 6888. Springer, 2011, pp. 148–163.

[5] P. Bonatti, S. D. C. di Vimercati, and P. Samarati, "An algebra for composing access control policies," *ACM Trans. on Inf. and Syst. Security*, vol. 5, no. 1, pp. 1–35, 2002.

[6] M. Harrison, W. Ruzzo, and J. Ullman, "Protection in operating systems," *Communications of the ACM*, vol. 19, pp. 461–471, 1976.

[7] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models," *IEEE Computer*, vol. 29, no. 2, pp. 38–47, 1996.

[8] R. S. Sandhu, "Lattice-based access control models," *IEEE Computer*, vol. 26, no. 11, pp. 9–19, 1993.

[9] D. Bell and L. LaPadula, "Secure Computer Systems: a Mathematical Model," MITRE, Tech. Rep. MTR-2547 (Vol. II), 1973.

[10] R. S. Sandhu, "On five definitions of data integrity," in *Proc. of the IFIP WG11.3 Working Conference on Database Security*, ser. IFIP Transactions, vol. A-47, 1993, pp. 257–267.

[11] S. Geller, C. Hauser, F. Tronel, and V. Viet Triem Tong, "Information flow control for intrusion detection derived from mac policy," in *IEEE International Conference on Communications (ICC'11)*, 2011.

[12] G. Hiet, V. Viet Triem Tong, L. Mé, and B. Morin, "Policy-based intrusion detection in web applications by monitoring java information flows," in *3nd International Conference on Risks and Security of Internet and Systems (CRiSIS 2008)*, 2008.

[13] V. Viet Triem Tong, A. Clark, and L. Mé, "Specifying and enforcing a fined-grained information flow policy : Model and experiments," in *Journal of Wireless Mobile Networks, Ubiquitous Computing and Dependable Applications (JOWUA)*, 2010.

[14] J. Zimmermann, L. Mé, and C. Bidan, "An improved reference flow control model for policy-based intrusion detection," in *Proceedings of the 8th European Symposium on Research in Computer Security (ESORICS)*, 2003.

[15] J. Barkley, "Comparing simple role based access control models and access control lists," in *ACM Workshop on Role-Based Access Control*, 1997, pp. 127–132.

[16] S. L. Osborn, R. S. Sandhu, and Q. Munawer, "Configuring role-based access control to enforce mandatory and discretionary access control policies," *ACM Transactions on Information and System Security*, vol. 3, no. 2, pp. 85–106, 2000.

[17] R. S. Sandhu, "A lattice interpretation of the Chinese Wall policy," in *Proceedings of the 15th NIST-NCSC National Computer Security Conference*, 1992, pp. 329–339.

[18] L. Habib, M. Jaume, and C. Morisset, "Formal definition and comparison of access control models," *J. of Information Assurance and Security*, vol. 4, no. 4, pp. 372–381, 2009.

[19] A. Chander, J. Mitchell, and D. Dean, "A state-transition model of trust management and access control," in *Proceedings of the 14th IEEE Computer Security Foundation Workshop CSFW*. IEEE Comp. Society Press, 2001, pp. 27–43.

[20] M. Tripunitara and N. Li, "Comparing the expressive power of access control models," in *11th ACM Conf. on Computer and Communications Security*, 2004.