# Personalizing Access Control by Generalizing Access Control

Steve Barker
Dept. Computer Science, King's College London
The Strand, London, WC2A 2LS
steve.barker@kcl.ac.uk

## ABSTRACT

We address the problem of providing data subjects with self-selected controls on access to their personal information. Existing approaches for this are not always sufficient in terms of offering the degrees of control and scope for individualization of access policies that are needed for personal data protection (and usage). We introduce a conceptual framework, a syntax, a semantics, and an axiomatization of a generalized form of access control meta-model, which may be specialized in various ways to enable data subjects to specify flexibly what access controls are to apply on their personal data.

## Categories and Subject Descriptors

D.4.6 [**Security and Protection**]: Access Controls.; D.2.8 [**Software Engineering**]: Metrics—*Logic, Security*

## General Terms

Security. Theory.

## Keywords

Access Control Models, Integrity, Privacy Policies

## 1. INTRODUCTION

The emphasis in access control has been primarily on organizations determining, specifying, maintaining and enforcing policies for controlling access to the "organization's" data. There are, however, a number of applications that require individual (especially human) entities to be able to choose flexibly what of their data should be accessible to whom, for what purpose, and in what circumstances.

The idea of entities having more control over the release of elements of their medical information (e.g., for health insurance quotes) to selected recipients has received attention (see, for example, [26]) and the idea of entities having control on access to "their data" when it is stored in e-form has even been viewed as an (inalienable) right [28]. A key research question thus arises: how can entities be provided with adequate means to enable them to define the often highly subject-specific access control policies that they may require to hold on $their$ data?

The closely related problem of helping to preserve the privacy of an entity's personal data has recently received attention (see, for example, the work on P3P [18], EPAL [5], Hippocratic databases [21], and XACML [2]), and several researchers in the access control community have proposed various "privacy-aware" access control models [23, 22]. Although these approaches allow data subjects[1] to express some controls on access to their personal information, we argue that, to differing extents, they do not provide sufficient expressive power for individually tailored access policies (e.g., typically they offer only limited opt-in/opt-out choices) and they fail to accommodate adequately several elements of access control (e.g., trust, delegation, meta-policy specification, and multi-policy specification). We also argue that, to varying degrees, these existing proposals have shortcomings in terms of at least one of the key criteria of: providing a shared conceptual view of core access control concepts, providing a common, shareable syntax, having an axiomatic base, and having a well-defined declarative semantics.

The limited expressive power that is offered by existing proposals for personal data protection, compromises the flexibility that data subjects have for specifying the precise access controls they wish to apply on "their" data. As such, existing approaches fail to satisfy adequately Westin's much quoted requirement, for privacy, that agents must be able "to choose freely under what circumstances and to what extent they will expose themselves, their attitudes, and their behavior, to others" [29].

Our focus is on providing data subjects with means for defining access controls on their data, to help to ensure its confidentiality and so that the data may be used for the data subject's personal advantage. We regard privacy as essentially a confidentiality-preserving problem that demands that extensions to existing access control models be developed.

The contributions that we describe in this paper may be summarized thus. We introduce a novel form of general (and thus commonly applicable), abstract meta-model of access control that can be specialized in multiple ways. We define a syntax, a semantics and an axiomatization for the model, which may be shared by data subjects, data controllers and data requesters. All of the models and policies that are expressed in terms of the syntax that we introduce have a common logical semantics and a common semantics is given to certain predicates that are included in the language that we introduce for defining our meta-model. Our key motivations are to provide shared concepts, syntax and semantics to facilitate ac-

[1]Henceforth, we use the term data subject to refer to a human user that contributes personal data to a data controller, which manages the storage of that data. A data user requests access to a data subject's information held by a data controller.


*SACMAT'10,* June 9–11, 2010, Pittsburgh, Pennsylvania, USA.

cess control policy representation, negotiation and exchange. Another motivation is methodological: we regard access control as representable in terms of a finite set of "self-evident axioms" (cf. Spinoza's *Ethics*) and access control models and policies as particular theories derivable from the (general) axiomatization. Our approach may also be viewed as defining a meta-model of access control that may be specialized to derive particular access control models and policies. Our methodological position is to emphasize universal not particular concepts and with a view to avoiding the (counter-productive) development of the next 700 access control models and 700 privacy models (cf. [6]).

In this paper, the specific focus is on the use of our general approach for deriving specialized access control models for personal data protection and exploitation; the example policies that we develop for this purpose will be referred to as the *self-managed access control (SMAC)* policies.

SMAC policies provide data subjects with control over a variety of aspects of their personal data; in particular, data subjects may specify the purposes for which their data may be stored and used, and the recipients that may access this data. Purpose and "external recipients" are common features in privacy policies; retention is the third key component that is typically found in privacy policy specifications and is usually interpreted in terms of data storage times. However, we prefer to consider *contextual accessibility* criteria. We argue that personalized access control is a highly relative notion and that contextual accessibility is fundamentally important for providing a policy author with the flexibility to define various context-based constraints on access to their personal data (e.g., times at which personal information is to be made accessible). Contextual accessibility subsumes retention as data storage and allows for highly dynamic policies to be defined by data subjects for protecting their personal data. Another novel feature of our approach is that policies on access to personal data is facilitated in a distributed as well as a dynamic context. In particular, we allow data subjects to choose freely what sources of information are to be used to determine authorized access to their data. The term "policy author" is interpreted quite liberally by us as applying to both data controllers and data subjects. Data subjects may revise a data controller's policy on the release of their personal data or the data subject may define its own policy on the release of this data. We use logic (rather than languages like P3P and EPAL) for succinct policy specification and to facilitate policy review and compliance checking by theorem-proving.

Although we recognize their importance in self-managed access control, due to space constraints, we will not consider obligations, audit policies, and hierarchies of objects or of purposes. We assume that data is stored and transmitted securely and that sound methods of authentication of data subjects, controllers and recipients are employed.

The remainder of the discussion is organized thus. In Section 2, we describe basic technical notions. In Section 3, we describe the core relations and axioms of our meta-model of access control. In Section 4 and Section 5, we consider the representation of various privacy-enhanced access control models and policies in terms of our formal framework. In Section 6, we consider practical matters. In Section 7, we discuss related work. In Section 8, conclusions are drawn and further work is suggested.

## 2. TECHNICALITIES

In this section, we describe the language for formulating our meta-model and specialized instances of it. We only describe the basic syntax and semantic notions (the minimum details to make this paper self-contained in terms of formal details); we refer the reader to [9] for further information on the formal language. It is essential to note from the outset that the logic language is used as a meta-language to describe object-level access control concepts in a precise way. Our approach does not rely on our particular choice of meta-language for description.

The main sorts of constants in the (non-empty) universe of discourse that we assume are as follows:

- A countable set $\mathcal{C}$ of categories, where $c_0, c_1, \ldots$ are (strings) used to denote arbitrary category identifiers.
- A countable set $\mathcal{K}_{ds}$ of data subjects and a countable set $\mathcal{K}_{du}$ of data users (requesters for access) where $\kappa_0$, $\kappa_1$, $\ldots$ are used for (key) identification.
- A countable set $\mathcal{A}$ of named atomic *actions*, where $a_0$, $a_1$, $\ldots$ are (strings) used to denote arbitrary action identifiers.
- A countable set $\mathcal{R}$ of *resource identifiers*, where $r_0$, $r_1$, $\ldots$ denote arbitrary resources (e.g., process identifiers, IRIs); $r(t_1, \ldots, t_n)$ is an arbitrary $n$-place relation that represents an "information resource" where $t_i$ $(1 \leq i \leq n)$ is a term, a function, a constant or a variable. An $n$-ary relation may be uniquely identified from the combination of predicate symbol, arity and location e.g., its URL.
- A countable set $\mathcal{P}$ of *purposes*, where $p_0, p_1, \ldots$ are (strings) used to denote arbitrary purpose identifiers.
- A countable set of meta-policy identifiers; for example, $c$ (for closed policies), $o$ (for open policies), $do$ (for a denials override policy), $\ldots$
- A countable set $\mathcal{T}$ of *time points*, $\tau_0, \tau_1, \ldots$.
- A countable set $\mathcal{E}$ of *event identifiers*, $e_0, e_1, \ldots$

Informally, a category (a term which can, loosely speaking, be interpreted as being synonymous with, for example, a type, a sort, a class, a division, a domain) is any of several fundamental and distinct classes or groups to which entities may be assigned (cf. [6]). In addition to allowing arbitrary forms of categories, the language that we choose is also not fixed in terms of things like the specific actions or purposes that are admitted (for generality). It should be noted that categories are not restricted to representation as properties; relations may be relevant too (cf. "taller than" and "tall"). The categories of interest will be application-specific and are determined by usage (individual, community or universal) rather than by necessary and sufficient conditions.

Times and events are important in our language to treat entities that may change category assignments. On times, we adopt a one-dimensional, linear, discrete view of time; a total ordering of time points that is isomorphic to the natural numbers. In this paper, we represent times in *YYYYMMDD* format. Two special time points will be important: 0 denotes the start of time and $\infty$ is an arbitrary maximal future time. We omit details on our choice of signature, but we assume that various comparison operators exist on times $\{<, \leq, \geq, >\}$, with their usual interpretation (e.g., $t_1 \leq t_2$ iff time point $t_1$ is earlier than or the same time point as $t_2$), and that arithmetic operators may be applied on times, e.g., $\{+, -\}$ for relative times. It is important to note that we adopt a "first-order" view of time, but other temporal systems can be incorporated in the scheme that we describe (fibring [20] is entirely consistent with our aim of generality).

The access control models and policies that we will develop are expressed using identification-based logic program (IBLP) rules [9], which take the form:

$$A \leftarrow A_1 \hookleftarrow v_1, \ \ldots, \ A_m \hookleftarrow v_m, \ \mathit{not}\ A_{m+1} \hookleftarrow v_{m+1}, \\ \ldots, \ \mathit{not}\ A_{m+n} \hookleftarrow v_{m+n} \quad (m \geq 0, n \geq 0)$$

Here, the head $A$ and each $A_i$ ($1 \leq i \leq m+n$) are atoms, and each $\upsilon_i$ ($1 \leq i \leq m+n$) is a URI in $\mathcal{R}$. The condition $A_i \hookleftarrow \upsilon_i$ is to be understood as $A_i$ as defined at $\upsilon_i$. Where $\upsilon_i$ is omitted from a condition then the literal in the condition of a rule is defined in a local access control policy specification; conditions may be of the form $A_i \hookleftarrow X$ where $X$ is a variable that may be instantiated by a URI. In simple terms, the semantics of an IBLP rule is that $A$ is true if $A_i$ is true at $\upsilon_i$ ($1 \leq i \leq m$) and *not* $A_j$ is true at $\upsilon_j$ ($m+1 \leq j \leq m+n$), where *not* $A_j$ is true at $\upsilon_j$ if $A_j$ is not finitely provable at $\upsilon_j$.

Functions are restricted to those satisfying the bounded-term-size property [15]. Attention is restricted to finite structures. As is conventional, variables in rules appear in the upper case; constants are in the lower case. An access control policy is a finite set of (locally) stratified IBLP rules [9], which admits a (finite) categorical model-theoretic semantics. IBLP rules are expressed in terms of a small number of core relations, each of which has a fixed interpretation. Policy-specific predicates may be defined by adding non-logical axioms to the logical axioms that define our meta-model.

# 3. THE $\mathcal{M}^P$ MODEL

In this section, we describe the common conceptual interpretation of access control and the common syntax and semantics that are adopted in our proposed meta-model of access control.

We note that the point has previously been made [6] that multiple access control models can be expressed in terms of a small set of primitive concepts. In this paper, we develop a range of privacy models, or (better) "privacy enhanced" access control models, from the meta-model of access control, denoted by $\mathcal{M}$, that we first introduced in [6]. When $\mathcal{M}$ was first introduced, we argued that its core elements could be variously specialized to meet particular requirements. The privacy-based interpretation of $\mathcal{M}$, henceforth denoted by $\mathcal{M}^P$, is essentially one such specialization. The other key point to note at the outset is that our concern is to define an axiomatization of access control in its generality. From this axiomatic base, access control models and policies may be developed (as logical theories) to which operational semantics (e.g., proof methods) may be applied to derive authorizations (as theorems) and to prove properties of policies (as theorems).

For $\mathcal{M}^P$, we extend $\mathcal{M}$ to accommodate data subjects, data controllers, denials of access, the notion of purpose, contextual accessibility criteria and the flexible specification of permitted recipients of a data subject's personal data. For that, the following core (interpreted) relations of the $\mathcal{M}^P$ model (defined with respect to our many-sorted language) are used:

- $\mathcal{PCA}$, a 4-ary relation, $\mathcal{K}_{ds} \times \mathcal{K}_{du} \times \mathcal{C} \times \mathcal{P}$.
- $\mathcal{ARCA}$, a 5-ary relation, $\mathcal{K}_{ds} \times \mathcal{A} \times \mathcal{R} \times \mathcal{C} \times \mathcal{P}$.
- $\mathcal{ARCD}$, a 5-ary relation, $\mathcal{K}_{ds} \times \mathcal{A} \times \mathcal{R} \times \mathcal{C} \times \mathcal{P}$.
- $\mathcal{PAR}$, a 3-ary relation, $\mathcal{K}_{du} \times \mathcal{A} \times \mathcal{R}$.
- $\mathcal{PRM}$, a 3-ary relation, $\mathcal{K}_{ds} \times \mathcal{R} \times \mathcal{M}$.

The semantics of the n-ary tuples in $\mathcal{PCA}$, $\mathcal{ARCA}$, $\mathcal{ARCD}$, $\mathcal{PAR}$, and $\mathcal{PRM}$ are, respectively, defined thus:

- $(\kappa_{ds}, \kappa_{du}, c, p) \in \mathcal{PCA}$ iff a data user $\kappa_{du} \in \mathcal{K}_{du}$ is assigned to the category $c \in \mathcal{C}$ for the purpose $p \in \mathcal{P}$ according to the data subject $\kappa_{ds}$.
- $(\kappa_{ds}, a, r, c, p) \in \mathcal{ARCA}$ iff the permission $(a, r)$ is assigned to the category $c \in \mathcal{C}$ for the purpose $p \in \mathcal{P}$ according to the data subject $\kappa_{ds}$.
- $(\kappa_{ds}, a, r, c, p) \in \mathcal{ARCD}$ iff a the permission $(a, r)$ is denied to the category $c \in \mathcal{C}$ for the purpose $p \in \mathcal{P}$ according to the data subject $\kappa_{ds}$.
- $(\kappa_{du}, a, r) \in \mathcal{PAR}$ iff a data user $\kappa_{du} \in \mathcal{K}_{du}$ is authorized to perform the action $a \in \mathcal{A}$ on the resource $r \in \mathcal{R}$.
- $(\kappa_{ds}, r, m) \in \mathcal{PRM}$ iff the data subject $\kappa_{ds}$ "controls" access to the resource $r \in \mathcal{R}$ and $\kappa_{ds}$ asserts that the meta-policy $m \in \mathcal{M}$ applies to access on the resource $r$.

The semantics of the $pca$, $arca$, $arcd$ and $prm$ predicates can also be understood in terms of a "says" relation (cf. [1]). That is, if $\Pi \models pca(\kappa_{ds}, \kappa_{du}, c, p)$, where $\Pi$ is an access control policy specification, then the data subject $\kappa_{ds}$ "says" that the data user $\kappa_{du}$ is assigned to the category $c$ for the purpose $p$. Similarly, for $arca$ ($arcd$) if $\Pi \models arca(\kappa_{ds}, a, r, c, p)$ ($\Pi \models arcd(\kappa_{ds}, a, r, c, p)$) then the data subject $\kappa_{ds}$ "says" that the $a$ privilege on $r$ is assigned (denied) to the category $c$ for the purpose $p$. If $\Pi \models prm(\kappa_{ds}, r, m)$ then $\kappa_{ds}$ "controls" access to $r$ and $\kappa_{ds}$ "says" what meta-policy, defined in terms of $arca$ and $arcd$, is to apply to access on $r$.

The elements in the set $\mathcal{PAR}$ are defined in terms of $\mathcal{PRM}$, $\mathcal{PCA}$, and a specification of a particular meta-policy $m$, which itself is defined with respect to $\mathcal{ARCA}$ or $\mathcal{ARCD}$. The rules defining $par$ for different meta-policies (closed (c), open (o), and denials-override (do)) are:

$$par(K_{du}, A, R) \leftarrow prm(K_{ds}, R, c), pca(K_{ds}, K_{du}, C, P), arca(K_{ds}, A, R, C, P).$$
$$par(K_{du}, A, R) \leftarrow prm(K_{ds}, R, o), pca(K_{ds}, K_{du}, C, P), not\ arcd(K_{ds}, A, R, C, P).$$
$$par(K_{du}, A, R) \leftarrow prm(K_{ds}, R, do), pca(K_{ds}, K_{du}, C, P), arca(K_{ds}, A, R, C, P), not\ arcd(K_{ds}, A, R, C, P).$$

Other meta-policies may be freely defined, provided that they are expressed in terms of the core predicates of $\mathcal{M}^P$.

On the semantics of $par$, for $\Pi \models par(\kappa_{du}, a, r)$ the requirement is (informally) that there exists a data subject $\kappa_{ds}$ that "controls" access to resource $r$ and that therefore may specify that: a meta-policy $m$ applies on access to $r$, $\kappa_{du}$ is assigned to the category $c$ for the purpose $p$, and the $a$ access privilege on $r$ is assigned or denied (as appropriate to satisfy the meta-policy) to members of $c$ for the purpose $p$.

Purposes are important for a data subject to be able to express fine-grained, highly individual access control requirements for helping to protect "their" data. On this, it should be noted that although the concept of purpose is important in determining authorizations, there is no argument for purpose in the relation named $par$. The extension of $par$ is the set of authorization triples $(\kappa_{ds}, a, r)$ that represents the permissions $(a, r)$ that the data subject $\kappa_{ds}$ has. In access control, generally, when a data subject requests a permission, the subject does so because it has an intention to act or a purpose for acting. However, this intention is typically ignored. In privacy-oriented approaches, intention often needs to be made explicit. For example, in "privacy-enhanced RBAC" a member of a role *doctor* (say) may be viewed as having different authorizations depending on the different purposes a principal might have when acting in the *doctor* role (e.g., when treating a particular patient and when acting as a medical researcher). We ignore purpose in terms of requesters for access and argue that purpose specifications are relevant only in terms of the relationship between data subject and data controller: the data subject decides what of its data may be released by the data controller for what purpose.

For representing hierarchies of categories, the following definition is included as part of the axiomatization of $\mathcal{M}^P$ (where '_'

denotes an anonymous variable):

$$contains(C, C) \leftarrow dc(C, \_),$$
$$contains(C, C) \leftarrow dc(\_, C),$$
$$contains(C', C'') \leftarrow dc(C', C''),$$
$$contains(C', C'') \leftarrow dc(C', C'''), contains(C''', C'').$$

Authorization may then be defined in $\mathcal{M}^P$ terms thus:

$$par(K_{du}, A, R) \leftarrow prm(K_{ds}, R, c), pca(K_{ds}, K_{du}, C', P), contains(C, C'), arca(K_{ds}, A, R, C, P).$$

In this instance, a closed policy is specified as being enforced by all data subjects, and $contains$ is a definition of a partial ordering of categories that are elements in the transitive-reflexive closure of a "directly contains" ($dc$) relation on pairs of category identifiers $dc(c_i, c_j)$, such that: $\Pi \models dc(c_i, c_j)$ iff the category $c_i \in \mathcal{C}$ ($c_i \neq c_j$) is senior to the category $c_j \in \mathcal{C}$ in a category hierarchy defined in $\Pi$ and there is no category $c_k \in \mathcal{C}$ such that $[dc(c_i, c_k) \wedge dc(c_k, c_j)]$ holds where $c_k \neq c_i$ and $c_k \neq c_j$. Although the partial ordering of categories is often a feature of access control models, it should be clear that other relationships between categories may be easily defined within the $\mathcal{M}^P$ model.

A motivation for us developing $\mathcal{M}$ [6] was to provide core access control concepts that may be specialized for specific needs. In terms of $\mathcal{M}^P$, $\mathcal{ARCA}$ may be defined as a 6-ary relation $\mathcal{K}_{ds} \times \mathcal{A} \times \mathcal{R} \times \mathcal{C} \times \mathcal{P} \times \mathcal{T}$ to admit permission assignments at an instance of time $t \in \mathcal{T}$, $\mathcal{M}^{PT}$ say, and if an interval-based semantics is required then $\mathcal{ARCA}$ may, for instance, be defined as a 7-ary relation $\mathcal{K}_{ds} \times \mathcal{A} \times \mathcal{R} \times \mathcal{C} \times \mathcal{P} \times \mathcal{T}_1 \times \mathcal{T}_2$, $\mathcal{M}^{PI}$ say, where $[t_1, t_2]$ is the closed interval for which the tuple $(\kappa_{ds}, a, r, c, p)$ is in the extension of $arca$. Similar enhancements are possible to allow for spatial constraints, for example. It should also be noted that defining $arca$ and $arcd$ in terms of a range of modalities beyond the interpretation of "can" as permission, as described in [6], are possible within the $\mathcal{M}^P$ family of models. Moreover, access control constraints, like separation of "duties" [3], may be defined as statements of the form,

$$\bot \leftarrow A_1 \hookleftarrow v_1, \ldots, A_m \hookleftarrow v_m,$$
$$not\ A_{m+1} \hookleftarrow v_{m+1}, \ldots, not\ A_{m+n} \hookleftarrow v_{m+n},$$

which have the following semantics: a specification $\Pi$ (expressed in the context of the $\mathcal{M}^P$ model) violates its constraints iff $\Pi \models \bot$ (where $\bot$ is falsum).

In summary, our formalization may be understood in terms of a triple $\langle \Phi, \Gamma, \Lambda \rangle$ where $\Phi$ is a definition of $\mathcal{PAR}$, $\Gamma$ defines the relationships between categories, and $\Lambda$ is a set of constraints. Each of these elements are specialized from $\mathcal{M}^P$ to construct particular access control models and policies. We consider this next.

# 4. SPECIALIZING THE $\mathcal{M}^P$ MODEL

In this section, we describe the representation, in $\mathcal{M}^P$ terms, of a range of extended existing access control models for personalized access control. We start our discussion by considering the representation of privacy-enhanced RBAC in $\mathcal{M}^P$ terms.

Standard RBAC models [3] assume a single (limited) form of category: the role. In ANSI Hierarchical RBAC, role hierarchies are the only form of category-category relationships that are admitted. The axiom that defines authorization in hierarchical RBAC can be expressed thus (cf. [6]):

$$par(P, A, R) \leftarrow pca(P, C), contains(C, C'), arca(A, R, C').$$

In this instance, $contains$ is the definition of a partial order relationship between pairs of categories (here restricted to roles) and a principal $P$ has $A$ access on resource $R$ if $P$ is assigned to a category $C$ (a role) that inherits from a category $C'$ (a role) that is junior to $C'$ and such that the $A$ privilege on $R$ is assigned to $C'$.

To develop, in the context of $\mathcal{M}^P$, privacy-enhanced hierarchical RBAC with subject-specified access controls and purposes, the definition of authorizations ($par$) is expressed thus:

$$par(K_{du}, A, R) \leftarrow prm(K_{ds}, R, c), pca(K_{ds}, K_{du}, C), contains(C, C'), arca(K_{ds}, A, R, C').$$

That is, a data user (requester) $K_{du}$ has $A$ access on resource $R$ if a data subject $K_{ds}$, which controls access to personal data in $R$, says that $K_{du}$ is assigned to a category $C$ (in this case a role) that inherits the $A$ privilege on $R$, to which a closed meta-policy on access applies, from a category $C'$ (a role) that is junior to $C$. Only positive authorizations may be specified in ANSI Hierarchical RBAC and so a closed meta-policy is represented in the specification above. Moreover, $K_{ds}$ in ANSI RBAC is typically a security administrator as security administrators (rather than data subjects) formally specify an *organization's* access policy.

Some matters of significance should be noted at this point in the discussion. In ANSI Hierarchical RBAC, $prm$ is not required but only because RBAC adopts the implicit assumption that data subjects are not policy administrators; policy administration is the responsibility of an organization-selected security administrator (SA). It immediately follows from this that, in ANSI RBAC, it is not a data subject that "says" that principal-category and permission-category assignments hold; by default, these specifications are the responsibility of the SA acting for an organization and therefore there is no need to capture the "says" relation explicitly. ANSI Hierarchical RBAC is a special case of $\mathcal{M}^P$ in another sense: for ANSI Hierarchical RBAC, $pca$ and $arca$ simply need to be specialized to omit any reference to a data subject. Alternatively, $pca$, $arca$ and $prm$ may be used exactly as specified in the $\mathcal{M}^P$ model but with the SA as the data subject. These different views may be flexibly adopted as is convenient. Either way, "privacy" may be added to access control in a seamless manner via the $\mathcal{M}^P$ model. The $\mathcal{M}^P$ meta-model can also accommodate 'privacy-enhanced" integrity models and business rules policies. We illustrate some of these points in the remainder of this section.

To accommodate purpose with subject-specified access controls in status-based access control [9], the axioms of $\mathcal{M}^P$ may be simply specialized thus (with the above definition of $contains$ assumed, with $E$ denoting an event, and with $C$ in this case being a category that combines ascribed and action statuses):

$$par(K_{du}, A, R) \leftarrow prm(K_{ds}, R, c), pca(K_{ds}, K_{du}, C, P), contains(C, C'), arca(K_{ds}, A, R, C', P).$$
$$pca(K_{ds}, K_{du}, C, P') \leftarrow current\_time(T), happens(E, T_s), agent(E, K_{du}), act(E, A'),\ T_s < T, pca\_init(E, K_{du}, C, A', [T_s, T], P'), not\ ended\_pca(K_{du}, C, [T_s, T], P').$$
$$ended\_pca(K_{ds}, K_{du}, C, T_s, T, P') \leftarrow happens(E', T'), agent(E', K_{du}),\ act(E', A''), pca\_term(E', K_{du}, A'', C, [T_s, T], P'), T_s < T',\ T' \leq T.$$

SBAC can also be specialized to allow for multiple business rules policies [8] and, by using the $\mathcal{M}^P$ model, business rules with privacy features.

Although SBAC may be viewed as a general form of RBAC, MAC and DAC can be viewed as special cases of RBAC [24]. In terms of $\mathcal{M}^P$, a version of the Bell-LaPadula model [11], with purpose $P$ and subject-specified access controls, may be viewed as a restricted form of the purpose-based Hierarchical RBAC model, in

which $contains$ is as previously defined and with $par$ defined thus, where L1 and L2 are categories that are specialized as classification or clearance levels:

$$par(K_{du}, read, R) \leftarrow prm(K_{ds}, R, c), pca(K_{ds}, K_{du}, L2, P), \\ contains(L2, L1), arca(K_{ds}, read, R, L1, P).$$
$$par(K_{du}, write, R) \leftarrow prm(K_{ds}, R, c), pca(K_{ds}, K_{du}, L1, P), \\ contains(L1, L1), arca(K_{ds}, write, R, L1, P).$$

In this case, the containment relationship is an ordering of categories that are restricted to being defined on a common set of security classifications, for resources, and security clearances, for subjects. The $par$ definitions represent the rules "no read up" and "write only at the subject's classification level," for a specified purpose, which are the core axioms of strict MAC (as the latter term is interpreted in Bell-LaPadula terms).

Privacy-enhanced DAC models may also be defined in terms of our general definition of authorization, viz.

$$par(K_{du}, A, R) \leftarrow prm(K_{ds}, R, c), pca(K_{ds}, K_{du}, C, P), \\ contains(C, C'), arca(K_{ds}, A, R, C', P).$$

In this case, the $pca$ and $arca$ definitions are provided by $K_{ds}$ on the resource $R$, as $K_{ds}$ is the owner of $R$, and the category-category relationship is typically one of delegation ($contains(C, C')$ if $C$ contains all delegees that $C'$ contains). In this case, $K_{du}$ is able to exercise the $A$ privilege on $R$ if $K_{du}$ can inherit the $(A, R)$ permission from a delegator.

Trust-based models that are extended with purpose and data subject control over personal data may also be defined within $\mathcal{M}^P$. For that, a discrete trust category may be viewed as simply another form of category on which subject-specific constraints on access to personal information can be defined. History-based, personalized access control models may also be represented in $\mathcal{M}^P$. For example, once an event has occurred such that a data requester has accessed a data subject's personal data, the requester may be categorized as amongst the set of requesters that are prohibited from further access to the data subject's personal data (cf. Chinese Wall policies [14]).

Integrity models, extended with purposes and data subject specifications of personal access requirements, may be defined in terms of $\mathcal{M}^P$; for example, the read and write rules of a Biba-like (strict) integrity model may be defined thus:

$$par(K_{du}, read, R) \leftarrow prm(K_{ds}, R, c), pca(K_{ds}, K_{du}, L2, P), \\ contains(L1, L2), arca(K_{ds}, read, R, L1, P).$$
$$par(K_{du}, write, R) \leftarrow prm(K_{ds}, R, c), pca(K_{ds}, K_{du}, L1, P), \\ contains(L1, L2), arca(K_{ds}, write, R, L2, P).$$

A combined Biba with Bell-LaPadula model may be defined within $\mathcal{M}^P$ by simply combining axioms.

A task-based privacy model of the type described in [19] may be naturally derived from $\mathcal{M}^P$ such that a data subject $K_{du}$ has access to "private" data for the purpose $P$ if $K_{du}$ is required to perform a task $T$ for which $A$ privilege on $R$ may be inherited from a task $T'$ to which the $A$ privilege on $R$ has been assigned, to wit:

$$par(K_{du}, A, R) \leftarrow pca(K_{ds}, K_{du}, T, P), contains(T, T'), \\ arca(K_{ds}, A, R, T', P).$$

A number of access control models, that differ in terms of the constraints that they admit, may also be expressed with respect to the core relations of $\mathcal{M}^P$. For example, the specification,

$$\bot \leftarrow pca(K_{ds}, K_{du}, c, P), pca(K_{ds}, K_{du}, c, P'), P \neq P'.$$
$$\bot \leftarrow arca(K_{ds}, write, \rho_1, C, P), arca(K_{ds}, write, \rho_2, C, P).$$

represents that exactly one data user $K_{du}$ may be assigned by a data subject $K_{ds}$ to the category $c$ for a specific purpose (a "separation of categories" constraint) and that $write$ privilege on the pair of resources $(\rho_1, \rho_2)$ is impossible for all categories of data subjects and for all purposes (a "separation of privileges" constraint).

# 5. SMAC POLICIES IN $\mathcal{M}^P$

In the previous section, we described, in high-level terms, a number of "privacy enhanced" access control models that can be derived from $\mathcal{M}^P$. In this section, we give examples of SMAC policies as specialized forms of $\mathcal{M}^P$. These examples are chosen for *exposition* purposes and as examples that highlight functionality.

We first introduce a technical component: annotated IBLP rules. An IBLP rule $\rho$ may be annotated with $\Delta$ to represent that a data subject is permitted by the controller to delete or modify $\rho$; the annotation $\neg\Delta$ is used to specify that $\rho$ cannot be changed by a data subject. In the latter case, a data subject $\kappa_{ds}$ is still free to insert rules of $\kappa_{ds}$'s choosing, but, not surprisingly, only for data that refers to $\kappa_{ds}$. Annotated IBLP rules take one of two forms:

$$\Delta : A \leftarrow A_1 \hookleftarrow \upsilon_1, \ \ldots, \ A_m \hookleftarrow \upsilon_m, \ not\ A_{m+1} \hookleftarrow \upsilon_{m+1}, \\ \ldots, \ not\ A_{m+n} \hookleftarrow \upsilon_{m+n}.$$
$$\neg\Delta : A \leftarrow A_1 \hookleftarrow \upsilon_1, \ \ldots, \ A_m \hookleftarrow \upsilon_m, \ not\ A_{m+1} \hookleftarrow \upsilon_{m+1}, \\ \ldots, \ not\ A_{m+n} \hookleftarrow \upsilon_{m+n}.$$

Although we restrict attention to annotations of rules in this paper, it is possible to extend their use to relations and even to terms. It is also important to note that, as we are concerned about access controls on a data subject's personal data, we assume that the information resource to be accessed by data requesters will contain a personal identifier of a data subject to which the data refers. We also assume that the objects to be accessed are relational structures [17];[2] functions are used by us to represent these structures, but an equivalent relational form is, of course, possible by reducing the term $f(t_1, \ldots, t_n)$ to the sequence of terms $\langle f, t_1, \ldots, t_n \rangle$, for example.

EXAMPLE 1. *Consider the following policy of the CAL Medical Center (CALMC) on the confidentiality of patient data:*

> *For the purposes of operating on a patient, the patient's full medical history, which includes the patient's identifier, name, date-of-birth, and history of illnesses, can be seen by any member of the category surgeon ($sur$) for the purpose of operating ($op$). The patient's identifier, name, date of birth and diagnosed illnesses in the past six months may be disclosed to the category of non-surgical staff ($nss$) for the purpose of providing diagnostic support ($ds$). The $pca$ definitions used by CALMC are defined non-locally at $\upsilon_1$. A closed access control policy is to apply to the release of all data. The access control policy as it relates to data subjects generally is maintained by the CALMC administrator denoted by $\kappa_c$.*

*Suppose that the databases used by CALMC include an 8-place relation p (where p is short for patient) that is defined at $\upsilon_2$ and includes details of the patient's identifier, the patient's name, date of birth, illness, room number (at the medical center), contact number (at the medical center), time of admittance and time of discharge:*

$$p(Id, Name, DoB, Illness, Rm, Pno, Admit, Discharge).$$

*To represent their requirements, CALMC's policy on the release of patient information may be represented as a SMAC policy, which*

[2]This is a non-restrictive, assumption; objects could simply be file identifiers, process identifiers, ...

*is simply derived from the $\mathcal{M}^P$ model, thus:*

$$\neg\Delta : pca(\kappa_c, K_{du}, C, P) \leftarrow pca(\kappa_c, K_{du}, C, P) \hookleftarrow \upsilon_1.$$
$$\neg\Delta : arca(\kappa_c, read, p(K_{ds}, V, W, X, Y, Z, T, T'), sur, op) \leftarrow p(K_{ds}, V, W, X, Y, Z, T, T') \hookleftarrow \upsilon_2, T \geq 0, T' \leq \infty.$$
$$\neg\Delta : arca(\kappa_c, read, p(K_{ds}, V, W, X, Y, Z, T, T'), nss, ds) \leftarrow p(K_{ds}, V, W, X, \_, \_, \_, \_) \hookleftarrow \upsilon_2, current\_time(T''), month(T'', M), month(T, M'), M' \geq M - 6.$$
$$\neg\Delta : prm(\kappa_c, p(K_{ds}, V, W, X, Y, Z, T, T'), c).$$
$$\neg\Delta : par(K_{du}, read, R) \leftarrow prm(K_{ds}, R, c), pca(K_{ds}, K_{du}, C, P), arca(K_{ds}, read, R, C, P).$$

□

From the example above, it should be noted that $\kappa_c$ is the controller of CALMC's SMAC policy. If any data subject $\kappa_{ds}$ were free to change CALMC's policy then $\kappa_{ds}$ could deny access to data users that need to have information on $\kappa_{ds}$ in order to perform an action of benefit to $\kappa_{ds}$ (e.g., diagnosing $\kappa_{ds}$'s illness). Nevertheless, $\kappa_{ds}$ does have the freedom to add to CALMC's SMAC policy specification in order to represent personal requirements on the release of $\kappa_{ds}$'s data. The next example demonstrates this.

EXAMPLE 2. *Consider the wishes of the patient John in relation to CALMC's policy on the disclosure of patient information:*

> *I agree to the hospital's policy on the release of my personal information for the purpose of operating. However, I also wish some of this information to be accessible to the category of data users that I call $family$. Specifically, the category $family$ is defined by me (non-locally) at $\upsilon_3$ and I want members of $family$ to be able to access (and only access) my name, bedside phone number, and room number for the purpose of contacting me while I am in hospital (a purpose that I denote by $ct$, as shorthand for contact).*

*To capture John's individual access control requirements, John adds the following definitions:*

$$prm(\kappa_{john}, p(\kappa_{john}, V, W, X, Y, Z, T, T'), c).$$
$$arca(\kappa_{john}, read, p(\kappa_{john}, V, \_, \_, Y, Z, \_, \_), family, ct) \leftarrow p(\kappa_{john}, V, W, X, Y, Z, T, T') \hookleftarrow \upsilon_2.$$

*John then adds the following $pca$ definition to CALMC's policy to express his required access controls applicable to his contacts (where $f\_mbr$ is short for "family member"):*

$$pca(\kappa_{john}, K_{du}, family, ct) \leftarrow f\_mbr(\kappa_{john}, K_{du}) \hookleftarrow \upsilon_3.$$

□

Various access control features are exhibited in the example above. For instance, $sur$ is a role within a hospital and so $par$ here will be a specialization of the $\mathcal{M}^P$ model that relates to roles; discretionary access controls apply (i.e., John decides on assignment to the category $family$); and content-based access control is supported. Delegation to third-party agents is represented by allowing a data subject to specify their own trusted third-party sources of additional assertions. Restrictions on access to its personal data are defined by a data subject in terms of the constants and variables that appear in either the head or the body of a policy rule.

Consider next an example of the use of our approach that is pertinent in the context of access control for secure e-trading.

EXAMPLE 3. *Suppose that Goods4U's access policy on the confidentiality of customer transaction data is expressed thus:*

> *Our preferred policy is to store a complete history of each customer's purchase transactions (the items bought, the number bought and when); we retain this information indefinitely and make it available at all times to subsidiaries of our choosing for the purpose of future marketing ($fm$). Any company that we call a subsidiary is assigned to the category that we call $sub$. We assign subsidiaries, for the purpose $fm$, from the time at which the subsidiary is first approved by us. A closed meta-policy is to apply on all forms of data release by default.*

*The databases that are used by Goods4U include a 3-place relation $su$, for subsidiaries, and a history of customer transactions is recorded in a 4-place relation, $tr$:*

$$su(SubId, Name, Start).$$
$$tr(CustId, Item, Number, Purchase\_Time).$$

*We assume that the definitions of predicates in $su$ and $tr$ are, respectively, found at $\upsilon_6$, and $\upsilon_7$. The $pca$, $arca$ and $prm$ definitions are assumed to be stored locally.*

*To represent their preferred access policy, Goods4U have the following expression of requirements:*

$$\Delta : pca(\kappa_c, K_{du}, sub, fm) \leftarrow su(K_{du}, N, T') \hookleftarrow \upsilon_6, current\_time(T), T' =< T.$$
$$\Delta : arca(\kappa_c, read, tr(K_{ds}, Y, N, T), sub, purch) \leftarrow tr(K_{ds}, Y, N, T) \hookleftarrow \upsilon_7, T \geq 0, T \leq \infty.$$
$$\Delta : prm(\kappa_c, tr(K_{ds}, Y, N, T), c) \leftarrow subject(K_{ds}), K_{ds} \neq K_{du}, not\ prm(K_{ds}, tr(K_{ds}, Y, N, T), c).$$
$$\Delta : par(K_{du}, read, R) \leftarrow prm(K_{ds}, R, c), pca(K_{ds}, K_{du}, C, P), arca(K_{ds}, read, R, C, P).$$

*Next, suppose that Paul is a customer with Goods4U and prefers to define his own access controls on his transaction history. On that, suppose that the following policy issues arise for Paul on access to his data that is held by Goods4U:*

> *I will allow my purchase history to be accessed but only by your subsidiaries that have a Better Business Bureau (BBB) rating that is equal or greater than B. My purchase history can only be released to subsidiaries that satisfy my additional criteria on BBB certification, I will only allow access to my transaction data as it relates to the purchase of widgets and the number of widgets bought by me (as I am only interested in widget-related purchases and data users may want to know if I am a "major purchaser"). Moreover, I do not want any release of my transaction data to any subsidiary for $fm$ purposes if my stock level of widgets, as recorded in $stock(item, quantity)$ at $\upsilon_{50}$, is greater than 100 units and I will only release my transaction history since 2009/01/01 and only until 2010/03/31 (after which time I will not be making any widget-related purchases so there is no reason for my data to be accessible to any external recipients after this time).*

*Assuming that the binary relation $BBBgrade$ is stored at $\upsilon_8$, to represent his requirements, Paul's specialization of Goods4U's SMAC policy can be represented thus:*[3]

$$pca(\kappa_{paul}, K_{du}, sub, fm) \leftarrow pca(\kappa_c, K_{du}, sub, fm), BBBgrade(K_{du}, G) \hookleftarrow \upsilon_8, G \geq b.$$
$$arca(\kappa_{paul}, read, tr(\kappa_{paul}, widget, N, \_), sub, fm) \leftarrow tr(\kappa_{paul}, widget, N, T) \hookleftarrow \upsilon_7, T \geq 20090101, T \leq 20100331, stock(widget, Q) \hookleftarrow \upsilon_{50}, Q > 100.$$
$$prm(\kappa_{paul}, tr(\kappa_{paul}, Y, N, T), c).$$

□

Temporal accessibility constraints and conditions on access that are defined in terms of notions like stock levels allow for dynamic

[3]Strictly speaking, $\geq$ is a comparison operator on the set of integers $\mathbb{Z}$, but character codes can, of course, be suitably mapped to elements of $\mathbb{Z}$.

SMAC policies to be formulated on data release. What Paul regards as private may vary over time. SMAC policies can change automatically in response to events and without requiring explicit policy modification. Moreover, Paul freely specifies the sources of access control information *of his choosing* to define allowed forms of access to *his data* e.g., the use of $BBBgrade/2$.

Access controls may also be represented within the SMAC framework by delegation because a "speaks for" relation is accommodated (cf. [1]). On that, suppose that $\kappa_{brian}$ is willing for his transaction history to be released to Goods4U's subsidiaries for the purpose of "future marketing" if $\kappa_{paul}$ specifies that his transaction history can be so released. That is, $\kappa_{brian}$ permits $\kappa_{paul}$ to "speak for" him in terms of the assignment of principals to categories ($pca$). However, suppose that unlike $\kappa_{paul}$, $\kappa_{brian}$ does not require a temporal constraint on the release of his transaction history. The required specification is:

$$prm(\kappa_{brian}, tr(\kappa_{brian}, Y, N, T), c).$$
$$pca(\kappa_{brian}, K_{du}, sub, fm) \leftarrow pca(\kappa_{paul}, K_{du}, sub, fm).$$
$$arca(\kappa_{brian}, read, tr(\kappa_{brian}, Y, N, T), sub, fm) \leftarrow tr(\kappa_{brian}, Y, N, T) \hookleftarrow \upsilon_7.$$

The next example, highlights flexible, subject-specific policy specification.

EXAMPLE 4. *Suppose that, like $\kappa_{brian}$, George wants to adopt $\kappa_{paul}$'s definition of pca but wishes to impose controls on the release of his transaction data such that a denial override policy is to be enforced to capture George's requirement that all of his transaction history may be released to whichever subsidiaries $\kappa_{paul}$ defines within Goods4U's policy but no history of George's purchase of bolts may be disclosed. George's preferences may be expressed by him as follows:*

$$prm(\kappa_{george}, tr(\kappa_{george}, Y, N, T), do).$$
$$pca(\kappa_{george}, K_{du}, sub, fm) \leftarrow pca(\kappa_{paul}, K_{du}, sub, fm).$$
$$arca(\kappa_{george}, read, tr(\kappa_{george}, Y, N, T), sub, fm) \; tr(\kappa_{george}, Y, N, T) \hookleftarrow \upsilon_7.$$
$$arcd(\kappa_{george}, read, tr(\kappa_{george}, bolt, N, T), sub, fm) \leftarrow tr(\kappa_{george}, bolt, N, T) \hookleftarrow \upsilon_7.$$
$$par(K_{du}, read, R) \leftarrow prm(K_{ds}, R, do), pca(K_{ds}, K_{du}, C, P), arca(K_{ds}, read, R, C, P), not\ arcd(K_{ds}, read, R, C, P).$$

□

That is, George uses a denial to define an exception to Goods4U's preferred policy on releases of transaction data in relation to his purchase of bolts; he also requires that a "denials override" meta-policy be enforced rather than Goods4U's closed policy. Notice too that when George adds his definition of *par* to Goods4U's policy, the combined policy allows for access if an authorization exists either under a closed meta-policy or a denial-overrides policy.

It is important to note that a data subject's policy on the release of personal data need not be more constrained than the controller's (for personalized control). The next example illustrates this.

EXAMPLE 5. *Consider the following policy rule on Goods4U's subsidiary-category assignments,*

$$pca(K_{ds}, K_{du}, sub, fm) \leftarrow su(K_{du}, N, T') \hookleftarrow \upsilon_6,$$
$$current\_time(T), T' \leq T, BBBgrade(K_{du}, a+) \hookleftarrow \upsilon_8.$$

*Rather that accepting this policy, the data subject Ringo may adopt the following policy rule to reflect his more liberal access conditions on Goods4U's subsidiaries:*

$$prm(\kappa_{ringo}, tr(\kappa_{ringo}, Y, N, T), c).$$
$$pca(\kappa_{ringo}, K_{du}, sub, fm) \leftarrow su(K_{du}, N, T') \hookleftarrow \upsilon_6, current\_time(T), T' \leq T.$$

□

As a final example, we demonstrate how multiple policies can be flexibly combined in $\mathcal{M}^P$.

EXAMPLE 6. *Consider a data subject Yoko, three arbitrary purposes $p_1, p_2$ and $p_3$, a role in the context of RBAC $\rho$, a status $\sigma$ in the context of SBAC, and a security clearance level $\lambda$. Next, suppose that Yoko has the following access control requirements (with $arca$ requirements omitted):*

> *I will allow any resource to be read by any data user that is a member of $\rho$ as defined by the data controller $\kappa_1$ at $\upsilon_\alpha$ for the purpose $p_1$, I will allow any resource to be read by any data user that is a member of $\sigma$ as defined by $\kappa_2$ at $\upsilon_\beta$ for the purpose $p_2$, and I will allow any resource to be read by any data user that is a member of $\lambda$ as defined by $\kappa_3$ at $\upsilon_\gamma$ for the purpose $p_3$. In all cases, a closed access policy is to be enforced.*

*To capture her (highly-specialized) requirements, Yoko defines the following SMAC policy (with the local $prm$ and $arca$ definitions omitted):*

$$pca(\kappa_{yoko}, K_{du}, \rho, p_1) \leftarrow pca(\kappa_1, K_{du}, \rho, p_1) \hookleftarrow \upsilon_\alpha.$$
$$pca(\kappa_{yoko}, K_{du}, \sigma, p_2) \leftarrow pca(\kappa_2, K_{du}, \sigma, p_2) \hookleftarrow \upsilon_\beta.$$
$$pca(\kappa_{yoko}, K_{du}, \lambda, p_3) \leftarrow pca(\kappa_3, K_{du}, \lambda, p_3) \hookleftarrow \upsilon_\gamma.$$
$$par(K_{du}, read, R) \leftarrow prm(\kappa_{yoko}, R, c), pca(\kappa_{yoko}, K_{du}, C, P), arca(\kappa_{yoko}, read, R, C, P).$$

□

# 6. PRACTICAL ISSUES

In this section, we briefly consider some practical issues relating to policy specification and enforcement in the context of $\mathcal{M}^P$.

We note first that the problem of determining whether an authorization holds with respect to any policy defined in $\mathcal{M}^P$ terms, without structured terms, is a $\Pi^1_1$ computation for the (locally stratified) IBLPs that define policies in our scheme. In practice, we believe that stratified IBLPs will be sufficient for many SMAC policies and other policies defined in terms of $\mathcal{M}^P$. On that, we note that the stable model of a reduced IBLP [9] with $n$ strata is a $\Sigma^n_0$ computation [4], and that an operational method exists for checking authorizations that is quadratic in the size of a configuration of IBLPs [27].

On an issue related to the complexity of deciding authorizations, two other matters need to be accounted for: the general expressiveness of the meta-model and how properties of policies that are defined in terms of $\mathcal{M}^P$ can be analyzed.

In simple terms, the expressiveness of $\mathcal{M}^P$ is quantifiable in terms of the meta-language that is used to define the object-level notions that relate to $\mathcal{M}^P$. The meta-language is the language of IBLPs with computations over a *configuration* of IBLPs [9]. The configuration reduces to a (locally) stratified logic program. Locally stratified programs are sufficiently expressive to allow the definition of all predicates that can be defined by programs with a unique stable model, which is precisely the $\Delta^1_1$ sets. Of course, specific access control models and policies, which may be represented in $\mathcal{M}^P$ terms, will often require less expressiveness than that required to define every hyperarithmetic set [12].

On the issue of proving properties of policies that are defined in terms of $\mathcal{M}^P$, we note that, as in the case of expressiveness, many properties of policies that are defined in terms of $\mathcal{M}^P$ are immediate consequences of the operational semantics that may be used for computation with respect to the meta-language used to define $\mathcal{M}^P$ (i.e., IBLPs). Establishing these properties is by proof.

Many properties of access control policies may be defined in terms of $\mathcal{M}^P$. For example, as the intended meaning of a configuration of IBLPs $\Sigma$ is its stable model and the stable model of $\Sigma$ is unique, it follows that policies defined in terms of $\mathcal{M}^P$ are

logically consistent. On the issue of inconsistency of permissions and denials (i.e., is any data subject assigned to any category $C$ permitted and denied the same permission for any purpose $P$?), it is a simple matter to define a constraint,

$$\bot_{+,-} \leftarrow arca(K_{ds}, A, R, C, P), arcd(K_{ds}, A, R, C, P)$$

on a policy $\Pi$ that is defined in terms of $\mathcal{M}^P$ and to determine whether $\bot_{+,-}$ is derivable (to decide whether it is the case that $\Pi \vdash arca(K_{ds}, A, R, C, P)$ and $\Pi \vdash arcd(K_{ds}, A, R, C, P)$ for some substitution for variables) by using a sound and complete method for theorem-proving with respect to IBLPs.

Properties like security and availability [10] may be interpreted in terms of the soundness and completeness of proof methods used for computation with respect to access control models and policies that are defined by (locally) stratified programs. A sound request evaluator $\varepsilon$ allows for provable security with respect to a model or policy $\Pi$ defined in terms of $\mathcal{M}^P$: $\Pi \vdash_\varepsilon par(\kappa_{du}, a, r) \rightarrow \Pi \models par(\kappa_{du}, a, r)$, where $\kappa_{du}, a$ and $r$ denote, respectively, an arbitrary data user, action and resource (i.e., every authorization provable from $\Pi$ by $\epsilon$ is a logical consequence of $\Pi$). The converse notion, of completeness, corresponds to availability: $\Pi \models par(\kappa_{du}, a, r) \rightarrow \Pi \vdash_\varepsilon par(\kappa_{du}, a, r)$ (i.e., every authorization that holds according to the logical specification $\Pi$ is provable from $\Pi$ by $\epsilon$).

Properties like category membership testing (is $\kappa_{du}$ assigned to category $c$?) and reachability analysis (is it possible for some data subject to access resource $r$ in order to write $r$?) are expressible in (locally) stratified logic and $\mathcal{M}^P$ terms. Such properties can therefore be expressed as candidate theorems for proof from access control models or policies, as logical theories, that are defined in terms of $\mathcal{M}^P$. In general terms, any property that is definable within a (locally) stratified configuration $\Sigma$ of IBLPs can be checked by using any sound and complete method for proof construction on $\Sigma$. Termination of operational methods for computing the stable model semantics provide a guarantee on the decidability of access request checking and property proving.

On compliance (or non-compliance) with policy requirements, a data subject $\kappa_{ds}$ is free to determine which recipients have what access to $\kappa_{ds}$'s data by evaluating $par$ requests with respect to a SMAC policy. From the results of a "failed" proof of compliance, $\kappa_{ds}$ may change the access controls on its data. We assume that data controllers may be trusted not to implement any policy that is hidden from $\kappa_{ds}$ and that leaks $\kappa_{ds}$'s personal information to a third party without $\kappa_{ds}$'s consent. We envisage data subjects querying a SMAC policy to extract information from the policy before choosing to agree to what it entails. Enabling data subjects to query SMAC policies and to understand their effects also provides a basis for policy negotiation.

On the practicality of our proposal, we note that SMAC policies may be implemented using Ciao Prolog [16] and in exactly the same way as we have described in previous work on benchmark testing of access control request evaluation [9]. As such, the attractive performance results reported in [9] can be expected to extend to implementations of SMAC policies. Moreover, the approach to policy specification that enables Jones Optimality to be achieved (cf. [7]) may be directly used with access requests on SMAC policies.

The logic language that we have used to define $\mathcal{M}^P$ has been chosen because it has a semantics that is common to a variety of practical languages that may be employed for implementation. For instance, many SMAC policies may be implemented in core SQL (with stratified negation) and for SMAC policy exchange, interoperability and policy enforcement on various rule-based processing platforms, RuleML may be used [13]. Our approach is technology-neutral; the implementation of policies derived from $\mathcal{M}^P$, like SMAC policies, is possible by using any method that computes the intended semantics of IBLPs. Our proposed approach is also policy neutral. That is, the $\mathcal{M}^P$ model allows for the flexible configuration of access control primitives for representing a wide range of access control model and policy requirements.

Access control models and policies that may be expressed in terms of $\mathcal{M}^P$ are expressed in high-level, declarative, logic terms. In relation to the Saltzer-Schroder security principles [25], allowing for a high-level, declarative representation of access control requirements and requiring only a small number of concepts in $\mathcal{M}^P$ is important for helping to satisfy the practically important criterion of psychological acceptability and the principle of economy of mechanism. Even though IBLPs provide a high-level, declarative language for policy specification it is, of course, possible to develop tools, front-ends, or a natural language representation that make it simpler still for data subjects to represent policy requirements in terms of $\mathcal{M}^P$.

# 7. RELATED WORK

The work that we have discussed in this article is related to that described in [6]. Our extended form of [6] includes features for accommodating: data subjects, data controllers, denials of access, the notion of purpose, contextual accessibility criteria, flexible meta-policy representation, and the flexible specification of external recipients of a data subject's personal data. Specific enhancements include the addition of the $prm$ relation for data subjects to choose what conflict resolution strategy to enforce on what of their data. The work in this paper also builds on [6] by demonstrating that the core concepts from [6] can be developed to include privacy as well as access control.

Some of the issues that arise in the context of self-managed access control have begun to be addressed in the work on P3P [18], EPAL [5], and Hippocratic databases [21]. There has also been some attempt to combine aspects of this work. One proposal has been to use P3P or EPAL for access control policy specification and to translate these specifications into policies that are implemented by using Hippocratic databases. In the ensuing discussion, we refer to this approach as the combined approach and we describe our proposal principally in relation to this approach.

The $\mathcal{M}^P$ meta-model and SMAC policies are based on a small number of core relations and a simple axiomatization, which admits a well-defined model-theoretic semantics. The combined approach does not have the same clean, simple, conceptual and formal basis. Hippocratic databases are not based on a satisfactory conceptual model of access control and several useful access control notions are missing e.g., denials of access and support for open policies. In addition, Hippocratic databases are an implementation-based approach rather than being access control model-based. Our proposal has been firmly grounded on well-defined access control concepts that have been expressed in a logic language for which formal semantics (declarative and operational) exist. The combined approach does not have a satisfactory underlying semantics; combining approaches is also likely to make the development of a satisfactory semantics difficult to achieve. It is already well known that the P3P proposal raises some troublesome semantic issues (so ambiguous and inconsistent P3P policies may be specified) and EPAL has an operational semantics that is dependent on rule order.

In the combined approach, there has been a failure to accommodate adequately several elements of access control e.g., trust, delegation (especially distributed delegation), meta-policy specification, and multi-policy specification. These elements are captured

in $\mathcal{M}^P$ and specializations of it, like SMAC policies. Moreover, unlike the combined approach, elements of integrity models and privacy models may be represented and freely combined in the $\mathcal{M}^P$ model and SMAC policies. By accommodating a range of access control features in $\mathcal{M}^P$, we argue that it becomes possible for data subjects to express more fully their individual requirements on access to their personal data. For example, meta-policy specification is important in the context of self-managed data because data subjects will often have different, individual requirements on the release of their data. For some data subjects, making their data available is key and so an open policy will be applicable; for other data subjects, security will be of paramount importance and so a closed policy will apply. In the SMAC specialization of $\mathcal{M}^P$, different meta-policies can apply to different data subjects for different purposes on different data resources to different categories of data users under different contextual conditions. This flexibility is not provided in the combined model and the scope it offers for individualization of access control policies is restricted as a consequence.

Amongst other issues, it is our view that P3P, EPAL and Hippocratic databases are essentially based on a "one-size fits all" assumption about policies (with limited opt-in/opt-out choices) and these approaches are too heavily focused on the privacy policies that an enterprise (as opposed to a data subject) wishes to adopt. Each of these features has the consequence of limiting the scope that data subjects have to express their individual access control requirements on their personal data. Moreover, each of P3P, EPAL and Hippocratic databases has its own distinct shortcomings. For example, in P3P the predefinition of things like data categories and purposes restricts the scope that exists for subject-specified access controls and Hippocratic databases are based on the assumption that SQL is to be used for policy enforcement. Furthermore, policy information is mixed in with ordinary data in Hippocratic databases. In our approach, data and policy are separated; in particular, there is no annotation of data with purpose. Our approach does not assume that a specific language will be used for implementation. Furthermore, we do not require that data requesters specify the reasons for accessing data (cf. request evaluation in Hippocratic databases); the data subject's preferences on release of their data are what matter, not the putative intentions the data requester has for accessing the data.

Our approach to policy authorship is also quite different to existing approaches; data subjects essentially decide on access to their data rather than requiring an often clumsy matching (or mismatching) of a data subject's preferences with what an enterprise wishes to enforce (with little scope over personal tailoring of policies to meet individual requirements). Our approach allows for dynamic policies on private data access to be effected autonomously and our logic language allows for executable specifications of access control policies. As such, policy enforcement is not separated from specification in our proposal (cf. P3P), and our $\mathcal{M}^P$ model provides more than just a syntax for policy representation (cf. XACML). We have developed an abstract model and determined its semantics and only then considered various languages that may be based on the semantics. Our approach also differs from some existing work in avoiding ad hoc, proprietary, implementation-based approaches for policy provision. Providing a shared, well-defined semantics allows for provable policy compliance in our proposal, a requirement that is not adequately satisfied in either the combined approach or XACML.[4]

The focus in P-RBAC [23] is, like much previously published work on access control and privacy, on enforcing "universally" applicable policy requirements enshrined in legislation (e.g., COPPA and GLBA). Our work is different in that it is based on an approach for representing individual, subject-specific, personal access policy requirements. P-RBAC is also based on an approach that combines access control and privacy. This integrated approach is desirable to support. Our approach is also integration-based. P-RBAC has the additional attraction of being based on a reasonably well-defined conceptual model (i.e., role-based access control [3]). Nevertheless, it is our view that enhancing a *particular* form of access control model for personal data protection introduces a problem that is common in existing work on supporting privacy policy formulation: the problem of unduly constraining the control that individual data subjects have for managing access to their data. Even though the notion of "role" can be given a quite general interpretation, "role" remains a particular instance of the more general notion of category [6] and category, being more general than "role", offers greater flexibility to data subjects defining access controls on their data.

A sceptic may argue that the emphasis in access control on particulars like "role" is important for simplification by abstraction. However, we argue that a focus on category as an access control primitive adds no significant conceptual complexity and actually simplifies access control by providing a basis for an abstract conceptualization of access control in its generality (thus eliminating the need for researchers to consider developing the next 700 access control models [6] and the next 700 privacy models).

A sceptic may also argue that what we have advocated is the use of a form of discretionary access control: a data subject is the "owner" of their data and, as such, the data subject decides which recipients have access to that data. However, although data subjects decide at their discretion what access controls are to apply on their data, these access controls may, in our approach, be defined in terms of any number of different access control models (and integrity or privacy models); the term "discretionary control" is more widely interpreted by us than its narrower use in the context of the existing literature on DAC.

# 8. CONCLUSIONS AND FURTHER WORK

Recall the research question that has been addressed in this paper: if users are to have more control over "their" data then how can they be provided with flexible means for defining the access policies that they require to hold on that data? The argument that we have presented on this may be summarized in the following way.

Paradoxically, we suggest that a possible solution to the problem of personalizing access control may be developed by generalizing access control (hence the title of the paper). To provide the necessary scope for self-managed access control, data subjects must first be provided with an understandable, general, abstract access control model that enables them to conceptualize notions easily. For that, we introduced the $\mathcal{M}^P$ meta-model (in Section 3, with the formal underpinnings outlined in Section 2). The $\mathcal{M}^P$ model can be specialized by data subjects in multiple ways so that it may be used to represent a range of access control models (see Section 4) and specialized policies, like SMAC policies (see Section 5), for personalized access control. The $\mathcal{M}^P$ model and SMAC policies have some attractive practical aspects associated with them (see Section 6), and we have tried to explain how our approach offers something different to existing related work (see Section 7).

On specifics, we note that the $\mathcal{M}^P$ meta-model is formally well-defined and is essentially based on the use of just five key interpreted relations (the $pra$, $pca$, $arca$, $arcd$ and $prm$ relations) and two proper axioms that define $par$ and $contains$. Application-

[4]We also regard the privacy profile proposed for Version 3.0 of XACML as too limited in expressive power to be useful in practice.

specific relations and non-logical axioms (expressible in IBLP terms) may be added to the core sets of $\mathcal{M}^P$ model features in order to enable data subjects to derive specific access control models and policies to satisfy their particular requirements on the protection and exploitation of their data. Providing data subjects with a simple, high-level, formally well-defined, implementation-independent, expressive framework for formulating their individual requirements on releases of their personal data and for checking preference compliance is a start towards addressing the key open question of how to provide means that might enable data subjects "to choose freely under what circumstances and to what extent they will expose themselves, their attitudes, and their behavior, to others" [29].

The main contribution of our work is to introduce the $\mathcal{M}^P$ metamodel and specializations of it for personalized access control (in the form of SMAC policies). However, another source of originality is methodological: rather than attempting to find specialized "solutions" to access control problems, we aim to identify and formally represent the foundational concepts of access control and to then construct general axiomatic frameworks from which specialized "solutions" to access control problems may be derived. We know of no other work that is based on developing a range of personalized access control policies from an access control metamodel, like $\mathcal{M}^P$. Another methodological point to note is that we have developed a conceptual model before attempting to construct a language for policy expression (cf. XACML and its retrofitting of an RBAC profile).

Future work includes to incorporate the notion of obligations and hierarchies of purposes in our model, to consider what additional modalities may need supporting (if any), to build in auditing procedures, to allow for fine-grained options on releases of personal data (e.g., releases only in aggregate or microdata form), to address the issue of stickiness of SMAC policies, to consider the role of negotiation and explanation in personalized access control, to consider how legislative requirements may be added as conditions in SMAC policy rules, to investigate the use of higher-order logics for the succinct representation of SMAC policies, to address the issue of developing simplified front-ends to shield users from the logic (if they require that) and to explore some important distinctions in personalized access control between rights, norms and self-interests.

# 9. REFERENCES


[1] M. Abadi, M. Burrows, B. W. Lampson, and G. D. Plotkin. A calculus for access control in distributed systems. *ACM Trans. Program. Lang. Syst.*, 15(4):706–734, 1993.

[2] A. H. Anderson. A comparison of two privacy policy languages: EPAL and XACMl. In *SWS*, pages 53–60, 2006.

[3] ANSI. RBAC, 2004. INCITS 359-2004.

[4] K. R. Apt and H. Blair. Arithmetic classification of perfect models of stratified programs. *Fundamenta Informaticae*, XIII:1–17, 1990.

[5] M. Backes, M. Dürmuth, and G. Karjoth. Unification in privacy policy evaluation - translating EPAL into Prolog. In *POLICY*, pages 185–188, 2004.

[6] S. Barker. The next 700 access control models or a unifying meta-model? In *SACMAT*, pages 187–196, 2009.

[7] S. Barker, M. Leuschel, and M. Varea. Efficient and flexible access control via jones-optimal logic program specialisation. *Higher-Order and Symbolic Computation*, 21(1):5–35, 2008.

[8] S. Barker and G. Lowen. Event-oriented web-based e-trading. *Electr. Notes Theor. Comput. Sci.*, 235:35–53, 2009.

[9] S. Barker, M. J. Sergot, and D. Wijesekera. Status-based access control. *ACM Trans. Inf. Syst. Secur.*, 12(1), 2008.

[10] S. Barker and P. Stuckey. Flexible access control policy specification with constraint logic programming. *ACM Trans. on Information and System Security*, 6(4):501–546, 2003.

[11] D. E. Bell and L. J. LaPadula. Secure computer system: Unified exposition and multics interpretation. *MITRE-2997*, 1976.

[12] H. A. Blair, V. W. Marek, and J. S. Schlipf. The expressiveness of locally stratified programs. *Ann. Math. Artif. Intell.*, 15(2):209–229, 1995.

[13] H. Boley, S. Tabet, and G. Wagner. Design rationale of ruleml: A markup language for semantic web rules. In *SWWS 2001*, pages 381–401, 2001.

[14] D. F. C. Brewer and M. J. Nash. The Chinese Wall security policy. In *IEEE Symposium on Security and Privacy*, pages 206–214, 1989.

[15] W. Chen and D. S. Warren. A goal-oriented approach to computing the well-founded semantics. *J. Log. Program.*, 17(2/3&4):279–300, 1993.

[16] *The Ciao Prolog System*, 2004.

[17] E. F. Codd. A relational model of data for large shared data banks. *Commun. ACM*, 13(6):377–387, 1970.

[18] L. F. Cranor. P3P : Making privacy policies more useful. *IEEE Security & Privacy*, 1(6):50–55, 2003.

[19] S. Fischer-Hubner. *IT-Security and Privacy*. Springer, 2001.

[20] D. M. Gabbay. Fibring logics. *Oxford University Press*, 1999.

[21] K. LeFevre, R. Agrawal, V. Ercegovac, R. Ramakrishnan, Y. Xu, and D. J. DeWitt. Limiting disclosure in hippocratic databases. In *VLDB*, pages 108–119, 2004.

[22] Q. Ni, E. Bertino, J. Lobo, and S. B. Calo. Privacy-aware role-based access control. *IEEE Security & Privacy*, 7(4):35–43, 2009.

[23] Q. Ni, A. Trombetta, E. Bertino, and J. Lobo. Privacy-aware role based access control. In *SACMAT*, pages 41–50, 2007.

[24] S. L. Osborn, R. S. Sandhu, and Q. Munawer. Configuring role-based access control to enforce mandatory and discretionary access control policies. *ACM Trans. Inf. Syst. Secur.*, 3(2):85–106, 2000.

[25] M. D. Schroeder and J. H. Saltzer. The protection of information in computer systems. *Procs. IEEE 63*, 9:1278–1308, 1975.

[26] W. Simons, K. Mandl, and I. Kohane. The PING personally controlled electronic medical record system: Technical architecture. *Journal of the American Medical Informatics Association*, 12(1):45–54, 2005.

[27] A. van Gelder. The alternating fixpoint of logic programs with negation. *J. Comput. Syst. Sci.*, 47(1):185–221, 1993.

[28] D. J. Weitzner, J. Hendler, T. Berners-lee, and D. Connolly. Creating the policy-aware web: Discretionary, rules-based access for the world wide web. In *in Elena Ferrari and Bhavani Thuraisingham, editors, Web and Information Security. IOS*. Idea Group Inc., 2005.

[29] A. Westin. *Privacy and Freedom*. New York: Atheneum, 1967.