

Privacy-aware RBAC - Leveraging RBAC for Privacy

Qun Ni
Purdue University, USA
ni@cs.purdue.edu

Elisa Bertino
Purdue University, USA
bertino@cs.purdue.edu

Jorge Lobo
IBM T.J. Watson, USA
jlobo@us.ibm.com

October 1, 2008

Abstract

In this article we introduce a model to enforce access control to data containing personally-identifiable information and as such privacy-sensitive. The model extends the well known RBAC models in order to express highly complex privacy-related policies, taking into account features like purposes and obligations. In the paper we discuss the notions of obligations and privacy-aware permissions. We believe that a full-fledged privacy-aware access control solution based on RBAC may have a great potential because it can be easily deployed in systems already adopting RBAC and would thus allow one to seamlessly integrate access control policies and privacy policies.

1 Introduction

Privacy is today a key issue in information technology and has received increasing attention from consumers, companies, researchers, legislators, and governments. Legislative acts in the United States, such as the Health Insurance Portability and Accountability Act (HIPAA) for healthcare and the Gramm Leach Bliley Act (GLBA) for financial institutions, require enterprises to protect the privacy of their customers¹. Although enterprises have adopted various strategies to protect customer privacy and to communicate their privacy policies to customers, such as publishing privacy policies on websites, or incorporating privacy seal programs (e.g. TRUSTe, ESRB, BBBOnline, and CPAWebTrust), these approaches do not provide systematic mechanisms to control how personal identifiable information is actually handled when and after it is collected. Privacy protection can only be achieved by enforcing privacy policies within an enterprise's online and offline data processing systems. Otherwise, enterprises' actual practices might intentionally or unintentionally violate the intended privacy policies.

Conventional access models, such as Mandatory Access Control, Discretionary Access Control, and Role Based Access Control (RBAC) [6, 14], are not designed to enforce privacy policies and barely meet privacy protection requirements, particularly, purpose binding (i.e. data collected for one purpose should not be used for another purpose without user consent), conditions and obligations. The significance of purposes, conditions, and obligations lies in the OECD Guidelines on the Protection of Privacy and Transborder Flows of Personal Data, in current privacy laws in many countries, and in public privacy policies of well known organizations, such as Google, IBM, and Amazon. The OECD guidelines are, to the best of our knowledge, the most well known set of private information protection principles on which many other guidelines, data-protection laws (e.g. EU Privacy Protection Directive, the Privacy Act and the Personal Information Protection and Electronic Documents Act in Canada), and many public privacy policies are based. Purposes are directly applied in the OECD *Data Quality Principle*, *Purpose Specification Principle*, and *Use Limitation Principle*.

¹We use the term *customer* to refer to an individual who releases personally identifiable information to an enterprise. We will also use the term "data subject" with the same meaning.

There are many situations in which access to private data imposes obligations, that is, actions that must be fulfilled at some point in time in order to perform an action on these data. For example, the OECD *Accountability Principle* states that “A data controller should be accountable for complying with measures which give effect to the principles stated above”. A common approach to implement this principle in operating systems or DBMS is to log each data access. Logging actions are obligations required by the majority of privacy policies.

Conditions, that is, prerequisites to be met before an action can be executed, are critical in some cases. One such case is related to children information. Rules in the Children’s Online Privacy Protection Act (COPPA) apply to the online collection of personal information from children under 13. Here “the age of the information source is under 13” is a condition that must be satisfied before the COPPA rules are applied.

Despite its limitations, existing access control technology can be used as a starting point for managing personal identifiable information [13]. Because both access control policies and privacy policies usually refer to the same set of data, they should not conflict, and using a single model for both kind of policies would greatly simplify management [2].

One approach to address the development of such an integrated model is to extend conventional access control models to support privacy-aware access control policies. By *privacy-aware access control policies* we refer to access control policies that incorporate additional information relevant for a controlled access to privacy-sensitive data. An example of such information is the intended use of the data. To facilitate the introduction of such a model in organizations, it is desirable that the model be defined as an extension of some widely deployed access control model. Thus we propose a family of Privacy-aware Role Based Access control (P-RBAC) models, naturally extending classical RBAC models to support privacy policies. Because RBAC is widely deployed in products and applications, we believe that by leveraging RBAC one can facilitate extending the access control policies currently deployed in organizations to support privacy-aware access control.

2 Related Work

Modern access control policy languages proposed by industry, like the OASIS eXtensible Access Control Markup Language (XACML) and the Enterprise Privacy Authorization Language (EPAL), mention the support of privacy policies as one of their objectives and are able to bind purposes with actions. However they handle obligations as abstract symbols without concrete elements, and therefore are unable to support direct translations of privacy laws into access control policies.

Over the years, various access control models have been proposed to address different security requirements. Among those models, RBAC [6] and the Usage Control (UCON) models [12] are the closest to P-RBAC. Because RBAC has been widely investigated and accepted by industry and has a relatively simple yet highly extensible structure, it is the basis for our privacy-aware access control model. Driven by requirements from digital rights management (DRM), UCON models, on the other hand, focus not only on the initial access control but also on a continuous usage control, and therefore have some similarities with P-RBAC, e.g., conditions and obligations. In some sense, the goal of DRM in UCON is similar to that of privacy protection in that content providers aim at controlling access and use of content also after its distribution. However, privacy protection differs from DRM with respect to its protection methodology. Privacy protection heavily depends on the access purpose and the fulfillment of obligations within time intervals, while DRM in UCON focuses on the accounting of accesses according to various criteria, e.g. membership-based metered payment and pay-per-use payment. Accordingly UCON models are focused on updating mutable attributes in obligations and do not provide any concrete temporal constraints in obligations. Therefore UCON only provides an abstract symbol, denoted by T, to represent a timer or an event. As we will discuss later, a more concrete temporal constraint model is required to precisely translate privacy laws into access

control policies. Moreover, no approach has been proposed for detecting conflicts and redundancies in UCON authorizations, obligations and conditions. These issues are addressed by P-RBAC.

One crucial component in a model aiming at expressing privacy policies is represented by the support for obligations. Several approaches to model and analyze obligations have been recently proposed. Bettini et al. [3] have formalized policies with obligations as Horn clauses and investigated the problem of choosing the best policy rules to minimize the provisions and obligations that a user has to fulfill in order to execute certain actions. Irwin et al. [7] have modeled the notion of obligation as a tuple of subject, action, objects, and a time window, and analyzed accountability of obligation fulfillments. Dougherty et al. [5] have proposed an abstract model that handles obligations as constraints on the program execution path. Compared to P-RBAC, the majority of those obligation models are high level abstract models and were not designed to satisfy the needs of privacy laws, and therefore lack support of crucial features, such as conditional obligations and repeating obligations. Moreover, none of these approaches are based on RBAC.

Hippocratic database systems [1, 8] have been proposed to address the problem of enforcing privacy policies, written by some standard languages like P3P and EPAL, in databases. Unlike P-RBAC that is a general, conceptual model, Hippocratic database systems, which are able to enforce both access control policies and privacy policies as well, are an ad-hoc, implementation-oriented approach. Hippocratic database systems and P-RBAC can however complement each other. Because the design of Hippocratic database systems was heavily influenced by P3P that is rather limited in expressing obligations, Hippocratic database systems are unable to deal with obligations commonly required by privacy laws. Hippocratic database systems support any conditions in policies that are expressible in SQL, but they do not provide techniques for detecting possible conflicts among policies. On the other hand, techniques proposed in the context of Hippocratic database systems, such as query modification, can be useful to enable database systems to enforce P-RBAC policies.

Compared to our previous work on P-RBAC [11, 10, 9], this article provides a comprehensive presentation of all the P-RBAC family models and summarizes techniques developed for policy analysis.

3 Desiderata

Before presenting P-RBAC, we discuss important guidelines concerning privacy policies that have been derived from the OECD principles and the privacy laws COPPA, GLBA, and HIPAA. P-RBAC has been designed based on those guidelines.

3.1 Purpose Binding

Purposes, which are bound to actions on personal identifiable information (*PII*), are required by the OECD *Data Quality Principle*, *Purpose Specification Principle*, and *Use Limitation Principle*. Purposes are widely used for specifying privacy rules in privacy acts and public policies. HIPAA rules clearly state purposes. The majority of public privacy documents posted at well known sites also specify purposes.

3.2 Condition and Obligation

Conditions are predicates that further limit the scope of an action, that is, they specify under which situation an action on some object is allowed. Conditions enable policy writers to specify fine-grained privacy policies that precisely meet requirements from privacy laws. Obligations regulate which actions a subject/**subjects** must fulfill at a certain time in order to allow a certain action to be performed at current time. To better understand why conditions and obligations are necessary components in privacy laws, let us look at some examples.

COPPA states that “Before collecting, using or disclosing personal information from a child, an operator must obtain verifiable parental consent from the child’s parent. This means that an operator must make reasonable efforts (taking into consideration available technology) to ensure that before personal information is collected from a child, a parent of the child receives a notice of the operator’s information practices and consents to those practices.” A permission to collect, use or disclose children information is conditional because verifiable parental consent must be confirmed. Furthermore, an obligation informing a parent and obtaining verifiable parental consent must have to be fulfilled before accessing children information if no parental consent has been obtained in the past. Thus such an obligation may be *conditional* as well. Once the obligations is fulfilled and a reply, either consent or rejection, is obtained from a specific parent, the parent should not be asked again the same question. Therefore before activating the obligation, it is important to check whether the consent has already been requested.

Another COPPA rule states that “At any time, a parent may revoke his/her consent, refuse to allow an operator to further use or collect their child’s personal information, and direct the operator to delete the information.” An interpretation of this rule from an access control point of view is that, once operators obtain parental consent and collect children information, they should immediately assign to the relevant parents permissions such as “revoke consent”, “refuse further use or collection”, or “request deletion”. One possible way to specify such permissions in formal access control policies is in the form of obligations that must be fulfilled without delay after children information collection. Therefore, obligations may include actions like “grant permissions” or “revoke permissions”.

One GLBA rule asserts that “Consumers are entitled to receive a privacy notice from a financial institution only if the company shares the consumers’ information with companies not affiliated with it, with some exceptions. Customers must receive a notice every year for as long as the customer relationship lasts.” There are two different obligations concerning consumers and customers. An obligation, “sending consumers a privacy notice”, should be fulfilled once after disclosing the consumers’ information. Another obligation, “sending customers privacy notices”, must be fulfilled periodically. At first glance, the latter one, “sending customers privacy notices”, seems not to be bound to any action, e.g. collecting, using, and disclosing, on customers’ information. However, such an obligation is related to an attribute of the information providers. The attribute specifies that these providers are not only consumers, but also customers². Therefore, the obligation is actually related to an action that changes a consumer to a customer.

In short, both conditions and obligations are necessary components when modeling privacy laws. Moreover, the above examples show that obligations are the most complex component. The relevant modeling features required to adequately express obligations can be summarized as follows:

- *Action binding*. Obligations are usually coupled with some action request ³, i.e., a subject commits himself to complete some obligations in order to execute a specific action on some objects. There are cases in which obligations are only connected with some special objects in policies. Nevertheless, a corresponding action can still be recognized in practice because usually there is an action that makes these objects special and thus is the action triggering these obligations.
- *Temporal constraints*. Temporal constraints are often included in obligations. Some obligations, referred to as *pre-obligations*, should be fulfilled before an action request is allowed and the result from the obligation fulfillment may affect the decision of the action request. Other obligations should be fulfilled after the action in the action request is executed, referred to as *post-obligations*. Intuitively,

²In the GLBA, a consumer is an individual who obtains or has obtained a financial product or service from a financial institution for personal, family or household reasons. A customer is a consumer with a continuing relationship with a financial institution.

³In the access control literature, the term “access request” is usually used instead of “action request”. However, in privacy policy, actions like “collect” and “disclose” are not an “access”, therefore we use a more appropriate term “action request” to replace “access request” thereafter in this paper.

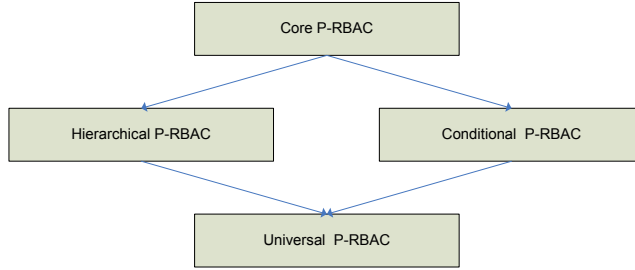


Figure 1: The family of conceptual P-RBAC models

there should be some time interval allocated for each obligation. Otherwise, a policy enforcement engine is not able to determine when it can start evaluating policy conditions if there are pre-obligations, and subjects can legally circumvent obligations by simply saying “I will do it in the future” in post-obligations. In some situations, temporal constraints require that obligations be performed repeatedly until some condition becomes true.

- *Different subjects.* A subject’s obligation may come from another subject’s action request, i.e., the subject of an obligation may be different from the subject who causes the obligation. For instance, when an operator discloses some children information to third parties, these parties may be required to fulfill obligations similar to the operator’s. In some situations, the subject of an obligation may be the system itself, e.g., logging accesses.
- *Conditional obligations.* Some obligations may be conditional, that is, they must be performed only if some condition becomes true. For instance, COPPA specifies that “An operator is required to send a new notice and request for consent to parents if there are material changes in the collection, use or disclosure practices to which the parent had previously agreed.” The “material changes” here are the prerequisites to execute the obligations “send a new notice” and “request for consent”.

4 Privacy-aware RBAC Family Models

Privacy-aware RBAC (P-RBAC) has been designed to support (possibly complex) privacy policies according to the guidelines discussed in the previous sections, and like classical RBAC consists of a family of conceptual models (see Figure 1) characterized by different modeling capabilities. Core P-RBAC, the base model, provides the basic components that are incorporated in advanced models such as Hierarchical P-RBAC and Conditional P-RBAC. There is a trade-off in the design of Core P-RBAC. On the one hand, Core P-RBAC should be sufficiently expressive to represent public privacy policies, privacy statements and privacy notices posted at Web sites, and policies based on privacy related acts. On the other hand, policy analysis in Core P-RBAC should remain tractable. Core P-RBAC is based on Core RBAC [6] without the session component and is discussed in detail in the following section.

Advanced models extend Core P-RBAC with additional modeling constructs. Hierarchical P-RBAC introduces the notions of *Role Hierarchy*(RH), *Object Hierarchy*(OH), and *Purpose Hierarchy*(PH); it thus enhances Core P-RBAC with hierarchical organizations for three important entities of Core P-RBAC. RH has the same notation and semantics as in Hierarchical RBAC [6]. OH describes a partial order relation (\leq) between different objects. The semantics of \leq relation is to propagate permissions among objects, i.e., given two objects $O_1 \leq O_2$, if a permission defines some action on O_2 , this permission implicitly defines the action

on O_1 as well. An example of OH is represented by an XML document; if a permission specifies “read an element e ”, the permission implicitly allows “read” on all sub-elements under element e as well. PH describes a partial order relation (\leq) between different purposes. The semantics of the \leq is to group purposes when defining permissions.

Conditional P-RBAC [10] introduces *Permission Assignment Sets* and *Complex Boolean Expressions* [10]. Conditional P-RBAC is able to express more complex conditions than those supported by the simple condition language of Core P-RBAC. New types of context variables, like string and integer, are supported in Conditional P-RBAC, as well as commonly used logical operators like negation and disjunction. Moreover, Conditional P-RBAC allows policy writers to specify relations between different permission assignments. The need of richer conditions is easily understood, but the importance of relations between permission assignments requires further explanation.

In RBAC, permissions are (action, object) tuples. For any access request, there may be several applicable permission assignments. By applicable we mean that the permission assignments contain the same action and object in the access request⁴. Permissions in these applicable permission assignments, however, must refer to the same tuple (action, object) of the request. Therefore, there is no need to consider the relation between several applicable permissions when answering the access request, e.g., should all applicable permission assignments be considered or only one of these permission assignments is sufficient. When conditions are introduced in permissions, the relation between applicable permissions becomes more complicated when answering an access request. If all applicable permission assignments must be considered and all conditions in these permissions must evaluate to true in order to allow the access request, the relation among these permission assignments is a conjunction (AND). If any of these permission assignments is sufficient and only one condition in these assignments needs to evaluate to true to allow the access request, the relation among the permission assignments is a disjunction. Because it is usually the case that all articles in a privacy act should be enforced, Core P-RBAC only supports the conjunction relation between permission assignments. It is obvious that such a design choice lacks flexibility and cannot handle situations requiring the disjunction relation⁵. To address such requirement, Conditional P-RBAC enables policy writers to specify both conjunctions and disjunctions among permission assignments.

Universal P-RBAC combines the features of both Conditional P-RBAC and Hierarchical P-RBAC and in addition provides three important features, namely, negative permissions, flow control of obligation execution, and permission combination principles. Negative permissions are neither supported by RBAC’96 [14] nor by the RBAC NIST Standard [6]. However, they are useful or even necessary in some situations. For instance, Article 8(1) of the EU Privacy Protection Directive requires that “Member States shall prohibit the processing of personal data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, trade-union membership, and the processing of data concerning health or sex life.” By enabling the translation of Article 8(1) directly into negative permissions one can avoid the additional steps required for representing and maintaining such a policy using only positive permissions. These steps are not trivial and must be proved to be correct. Having both positive and negative permissions has its own problems since it can result in “inconsistent” policy sets. Permission combination principles are introduced to address possible inconsistencies.

Due to space limitations, the following sections focus on Core P-RBAC. It is important to notice that some fundamental concepts of the RBAC standard [6], such as sessions and constraints, can be seamlessly added to in P-RBAC. We do not discuss them here because the focus of our discussion is on how to express privacy policies in P-RBAC.

⁴Roles in applicable permission assignments must be assigned to the user in the access request.

⁵Some disjunctive cases can be handled by using splitting context variables [11].

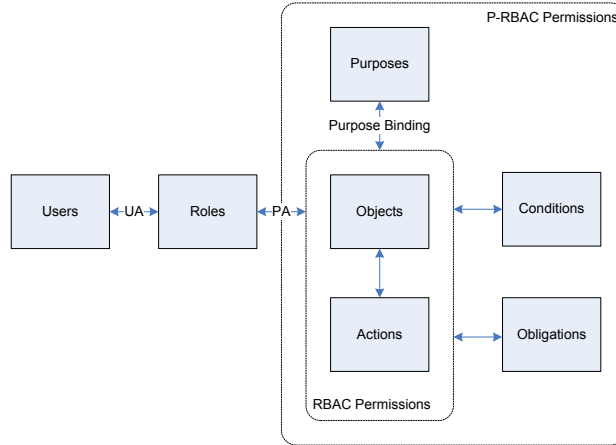


Figure 2: Core P-RBAC model

5 Core P-RBAC

Core P-RBAC (see Figure 2) includes several sets of entities: Users(U), Roles(R), Objects(O), Actions(A), Purposes(Pu), Obligations(Ob), and conditions (C) expressed by using a customized language, referred to as LC_0 . A user is human being, and a role represents a job function or job title within the organization with some associated semantics regarding the authority and responsibility conferred on a member of the role [6]. Note that in some sense, individuals external to the organization collecting the data, such as the parents of children whose information has been collected, can be considered a job function as well, because from the the organization perspective, these individuals have some authority of the data. In the case of the parents, it is clear that they have some authority regarding their children information. An object refers to any resource that should be protected. An action is an executable image of a program, which upon invocation executes some function for the user. The types of actions and objects that P-RBAC controls depend on the type of system in which P-RBAC is deployed.

It is important to point out that roles in P-RBAC not only represent groups of individuals within an organization but also groups of data subjects. Someone may argue that “Privacy is understood as relating to individuals whereas roles are concerned not with individuals but with categories of individuals. Thus, the perspectives of privacy and RBAC are different and from this potential tensions arise.”⁶ We believe that just like the situation in which roles are helpful for simplifying the administration of access control policies they can also be helpful for the administration of privacy policies. In P-RBAC we thus deal with two classes of individuals: the individuals represented by “data subjects” about whom data is collected; and the individuals that need to access the data collected about the data subjects. Accesses by the individuals in the latter class must be controlled to make sure that privacy regulations are not violated. Accesses to data by the individuals in the former class should be allowed in a limited form as well to comply with privacy laws, for example for enabling data subjects to add, delete, or change their own information. From the point of view of access control policy administration, there is thus no major difference between assigning roles to data subjects and assigning roles to data users. Personal preferences concerning privacy protection are not encoded in roles but in attributes about data subjects collected as privacy meta-data that can be accessed by context variables in permissions.

⁶This opinion was kindly offered by one of reviewers of this article.

In Core P-RBAC, as in classical RBAC, permissions are assigned to roles and users obtain such permissions by being assigned to roles. The distinctive feature of Core P-RBAC lies in the complex structure of privacy permissions, which reflects the highly structured ways of expressing privacy rules to represent the essence of OECD principles and privacy acts. Therefore, aside from the data and the action to be performed on it, a privacy permission explicitly states the intended purpose, along with the conditions under which the permission can be granted, and the obligations that are to be performed if permission is granted.

5.1 Conditions

Core P-RBAC conditions should not be confused with constraints in the classic RBAC model. Constraints are a mechanism for laying out higher-level organizational policies that cross over several roles, while conditions are a mechanism to precisely define a permission of a single role. A common example of constraints is separation of duties. Handling separately privacy-related conditions and constraints allows us to focus on how to effectively and precisely model the necessary prerequisites for enforcing privacy policies.

Core P-RBAC includes a simple language, LC_0 , for expressing conditions; they are expressed using *context variables*. Such variables record privacy-relevant information that needs to be considered when enforcing privacy permissions. Even though the LC_0 condition language has limited expressive power, it is able to model many conditions usually found in privacy acts and privacy policies. The conditions that can be expressed by LC_0 are defined in what follows.

Definition 5.1. [11] Let CV be a set of context variables; each variable $x \in CV$ has a finite domain of possible values, denoted as D_x ; every domain is equipped with a pair of corresponding relational operators $=$ and \neq . \perp (false) and \top (true) are constant conditions. An atomic condition ac defined over CV has the form $(x \text{ op}_r v)$ where $x \in CV$, $v \in D_x$, $\text{op}_r \in \{=, \neq\}$. The conditions of LC_0 (over CV) are defined as follows:

- A constant condition is a condition of LC_0 .
- An atomic condition is a condition of LC_0 .
- Let c_i and c_j be conditions of LC_0 . $c_i \wedge c_j$ is a condition of LC_0 .

□

Two typical context variables are:

- oc : it represents the data subject’s consent for collecting, using, or disclosing his personal identifiable information; its domain is $\{\text{yes, no, na}\}$. “na” means ‘not available’ and represents the situation in which the data subject has not been asked for the consent.
- vpc : it represents a parent’s consent for collecting, using, or disclosing his child information; its domain is $\{\text{yes, no, na}\}$.

Since the set of context variables is finite, each context variable has a finite domain, and only equality and inequality are supported, it is straightforward to see that the set of all possible conditions (module logical equivalence) expressed in LC_0 is finite; C denotes this set.

5.2 Obligations

As we mentioned in the previous section, the subject of an obligation may differ from the user who performed the action that has triggered the obligation; therefore a component used to indicate the obligation subject is also added to the permission. This should not be interpreted as to giving a blank permission to the obligation subject to execute the action imposed by the obligation. The obligation subject must independently have

Table 1: The ICV Set

Name	Domain	Representing
self	the set of users	the user who submits the action request
auser	the set of users	a possible user
ra	the set of roles	the role of the user whom the variable auser represents
users	the powerset of the set of users	all possible users
rs	the set of roles	the role of users whom the variable users represents

permission to execute the action during the time frame that the obligation must be fulfilled; otherwise the obligation will be violated.

It is quite common that, when defining a permission assignment, the obligation subject is not fully identified. In some cases, such a subject is the user that submitted the action request. There are other cases in which the subject is chosen from the set of users assigned to a particular role. The latter case is common in privacy laws because subjects in law statements usually refer to groups of individuals that share some properties. In these cases the P-RBAC permissions will explicitly identify the obligation subjects using some special context variables (see Table 1). These special context variables form a subset of CV , and are mainly used in conditions to refer to the obligation subjects.

Our temporal constraint model is based on a simple notion of time domain, that is, the pair $(\mathbb{Z}; \leq)$. In our context, each element of \mathbb{Z} is referred to as a *time instant* and \leq is a total order on \mathbb{Z} . In what follows, given $t, t' \in \mathbb{Z}$, $[t, t']$ denotes the time interval starting at time instant t and ending at time instant t' . Next definition introduces a terse yet flexible definition for temporal constraints which is the key notion in our temporal constraint model.

Definition 5.2. A *temporal constraint* tc is a tuple (t_s, t_e, cnt) , where $t_s, t_e \in \mathbb{Z}$, and $cnt \in \mathbb{N}^*$. t_s and t_e represent a time interval starting from t_s and ending at t_e and cnt represents the repeating number of the time interval. tc denotes a sequence of time intervals defined as follows:

- $[t_s, t_e], [t_e + 1, 2t_e - t_s + 1], \dots, [t_s + (cnt - 1)(t_e - t_s + 1), t_e + (cnt - 1)(t_e - t_s + 1)]$ if $t_e \geq t_s \geq 0$;
- $[t_s - (cnt - 1)(t_e - t_s + 1), t_e - (cnt - 1)(t_e - t_s + 1)], \dots, [2t_s - t_e + 1, t_s - 1], [t_s, t_e]$ if $0 \geq t_e \geq t_s$.

□

For instance, a temporal constraint $(3, 7, 3)$ represents the following sequence of time intervals: $[3, 7], [8, 12], [9, 13]$. For simplicity, we further assume in the P-RBAC model that time is relative, and we model it as an integer representing the number of basic time instants from a predetermined time instant, denoted by 0, which is related to an action request. We identify two special time instants that are related to the predetermined time instant 0: (1) a time instant, referred to as *decision time*, that is equal to the time instant at which the decision of granting the permission to execute the action is made; (2) a time instant, referred to as *completion time*, that is equal to the time instant of the completion of the action execution. For a time instant t , if $t > 0$, then t represents $|t|$ time instants after a completion time. If $t < 0$, then t represents $|t|$ (i.e. the absolute value) time instants before a decision time. For a temporal constraint (t_s, t_e, cnt) ⁷, if $t_e \leq 0$, the temporal constraint indicates that the executed obligation is a *pre-obligation* and should be fulfilled during the time interval between the request time and the decision time. If $t_s \geq 0$, the temporal constraint indicates that the associated obligation is a *post-obligation* that should be fulfilled sometime after completion time. As a special case, $t_e = t_s = 0$ indicates a post-obligation that should be fulfilled right after the action has been executed.

⁷It is obvious that $t_s \leq t_e$ must hold.

The distinction between the time before granting the permission to execute the action and the time after the execution of the action gives $t = 0$ a different meaning depending on whether t represents a starting time t_s or an ending time t_e . In classical access control policies without obligations, the time at which an action request occurs, the time at which to evaluate the condition of applicable permission assignments, the time at which to authorize the action, and the time when the action is performed, are considered to be the same. In practice however these times are different especially when obligations and temporal constraints are introduced.

In a temporal constraint, when $t_e = 0$ and $t_s < 0$, the constraint requires that the evaluation of the pre-obligation be completed before the evaluation of the permission condition can start. Such a requirement ensures that the execution of actions within the pre-obligation that affect the value of context variables in permission conditions is completed before the permission condition is evaluated. For instance, in the example policy “Before collecting, using or disclosing personal information from a child, an operator must obtain verifiable parental consent from the child’s parent.”, the obligation to “obtain verifiable parental consent” may affect a context variable vpc ’s value by assigning to it a value from the set { “yes”, “no” , and “na” }. The vpc ’s value in turn affects the evaluation of the conditions in the policy that are used to decide whether children information can be collected, used, or disclosed. $t_s = 0$ represents a different situation. If an action request is granted, it is possible that the subject who submits the request decides not to execute the action because of the burden imposed by the post-obligations. Therefore, it is reasonable that only after the action has been performed the timer recording the time interval allocated to post-obligations starts counting.

The following definition formalizes the obligation model adopted for P-RBAC.

Definition 5.3. Let C be a set of possible conditions expressed in LC_0 , U be a set of users, ICV be a set of special context variables, A be a set of actions, O be a set of objects, and TC be a set of temporal constraints. An obligation is a tuple (c, s, a, \bar{o}, tc) , where $c \in C$, $s \in U \cup ICV$, $a \in A$, $\bar{o} \in \mathfrak{P}(O)$ (a powerset of O), and $tc \in TC$. \square

U represents both human beings and any other subject who can initiate an action. To be clear, we call any object who can initiate an action a *user* in this paper. O includes any other component introduced in the definition, e.g. U , ICV , and TC etc. Moreover, data, not mentioned in the model, is a subset of O as well. It should be noted that in theory TC could be an infinite set. Nonetheless, it is reasonable for us to assume that for each deployment of Core P-RBAC there is only a finite set of temporal constraints applicable. Based on this assumption, we further conclude that the number of possible obligations is finite as well. Some obligation examples using context variables are as follows:

- $(c, self, a, \{o\}, tc)$: if condition c is true, the subject who submits the action request activating the obligation, performs the action a on object o under the temporal constraint tc .
- $(ra = r, auser, a, \{o\}, tc)$: a user with role r is obligated to perform the action a on object o under the temporal constraint tc .
- $(rs = r, users, a, \{o\}, tc)$: all users of role r are obligated to perform the action a on object o under the temporal constraint tc .

5.3 Core P-RBAC Model

The following definition combines all the previous notions and formally introduces core P-RBAC.

Definition 5.4. The Core P-RBAC model is composed of the following components:

- A set D of data, a set U of users, a set R of roles, a set Pu of purposes, a set A of actions, a condition language LC_0 , and a set O of objects such that $D \cup U \cup R \cup Pu \cup A \cup CV \subset O$.

- A finite set TC of temporal constraints defined according to Definition 5.2, and $TC \subset O$.
- A set Ob of obligations on O defined according to Definition 5.3.
- A set $P \subseteq C \times A \times \mathfrak{P}(O) \times \mathfrak{P}(Pu) \times \mathfrak{P}(Ob)$ of permissions.
- A set $PA \subseteq R \times P$ of permission assignments: a many-to-many permission to role assignment relation.
- A set $UA \subseteq U \times R$ of user assignments: a many-to-many user to role assignment relation.

□

Sessions, a component of the RBAC standard definition [6], are omitted here for simplicity. The concept of sessions could be integrated in our model, but its details and the analysis of the interaction between sessions and obligations are out of the scope of this article. As we can see, for any permission $p = (c, a, \bar{o}, \bar{p}_u, \bar{o}_b)$, p is a regular access control permission if $c = \top$, $\bar{p}_u = \emptyset$, and $\bar{o}_b = \emptyset$. Therefore, our Core P-RBAC permission set is a superset of the permission set in Core RBAC model.

5.4 An Example

We now discuss a rule from COPPA to show how privacy law rules with obligations can be represented in P-RBAC. **This rule, though it looks simple, includes several features supported by P-RBAC, such as repeating obligations, conditional obligations, and pre-obligations. Due to space limitation, we refer the reader to [9] for additional examples that illustrate other useful features such as a method that deals with obligations with subject binding instead of action binding.**

Policy 1(COPPA): “Before collecting, using or disclosing personal information from a child, an operator must obtain verifiable parental consent from the child’s parent.”. To represent such policy we must specify three permission assignments, each of which corresponds to one of three actions: collect, use, and disclose. Because they are similar, we only show the permission related to the collection. Such permission is as follows:

$$PA_1 : (\text{operator}, \text{vpc} = \text{yes}, \text{collect}, \{\text{ci}\}, \emptyset, \{(\text{vpc} = \text{na}, \text{self}, \text{obtain}, \{\text{vpc}, \text{pi}\}, (-\text{wd}, 0, 2))\})$$

where: vpc denotes the context variable “Verifiable Parental Consent”, ci denotes the object “children information”, and pi denotes the context variable “the parent whose child information is collected, used, or disclosed in permissions”. Because the original policy does not specify a purpose, the purpose component is an empty set. wd is a predefined constant number indicating the number of days to wait for the response. $(-\text{wd}, 0, 2)$ indicates that obtain is a pre-obligation and if there is no reply in wd days an operator can ask again. The value na indicates that the corresponding parent has never been asked for consent before. If the value of vpc is not na (that is, it is yes or no), it is reasonable that the parent should not be asked again.

5.5 Some Words about Policy Analysis

Large organizations usually have to deal with complex access control policies and privacy policies. The more complex these policies are, the higher is the possibility that policies contain errors, because of new requirements, new regulations, interaction among policies, or just human mistakes. Therefore, there is a need for techniques to detect incorrect policies before they are deployed. To address such issue, some primitive operations for analyzing P-RBAC policies have been identified, namely: validity of permission assignments, dominance of obligations, and indeterminism of obligation enforcement.

If obligations contained in a new permission contradict existing permissions or the permission itself, the new permission is not valid. The interaction between permissions and obligations may cause problems, such as for example, when there is no permission allowing the subject of an obligation to perform the action in

the obligation, or if the conjunction of the condition in the obligation and the condition in the permission containing the obligation is not satisfiable. Obligation execution may result in obligation cascading, that is, the execution of an obligation can trigger the execution of another obligation which in turn may trigger other obligations. Ill-written permissions can result in non-terminating obligation execution. The notion of cascading bag [9] has been proposed to address such an issue.

The dominance of obligation characterizes the different “strictness” between two obligations. Some requests may have several applicable policies that could lead to a large number of obligations. Therefore reducing the number of obligations to be executed may have significant practical impact. Obviously, the remaining obligations should not “reduce the duties” required by the original policies. On the other hand, we can expect many of these obligations to be similar to each other since they are obligations associated with similar permissions. An efficient algorithm for the minimization of obligation fulfillments, that takes into account temporal constraints, was presented in [9].

Indeterminism in obligation enforcement arises because multiple policies can apply to the same access request. Given a request to which two or more policies apply, the choice of the obligations to be executed can be non-deterministic if policies are not well written, e.g., the relationship between policies is a disjunction and the policies have different obligation sets. Policy normalization [10] has been proposed as an approach to detect indeterminism in obligation enforcement.

Techniques related to detecting conflicting policies and redundant policies in P-RBAC have been investigated as well, but, for space reasons, we omit such discussion and refer the interested reader to [11, 10].

6 Conclusions

In this article, we have presented the P-RBAC family models with various capabilities. We have elaborated on the motivations of Core P-RBAC through rules in privacy laws and explained relevant design rationale. We have also shown that Core P-RBAC can directly capture the semantics of privacy laws and we have discussed the main requirements of an obligation model.

Many interesting problems are however still left open. The interaction between obligations and permissions is an interesting topic further complicated if more expressive condition languages, delegation models, or hierarchies are introduced. An obligation model with more flexible flow control of obligation execution may be desired. For instance, causal relations may exist between obligations, and policies may have several alternative choices of obligations. The development of refinement techniques for P-RBAC policies is an important issue to directly support and automate the translation of high level privacy-protection goals onto low level policies.

References

- [1] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Hippocratic databases. In *VLDB*, pages 143–154. Morgan Kaufmann, 2002.
- [2] A. H. Anderson. A comparison of two privacy policy languages: Epal and xacml. In A. Juels, E. Damiani, and A. Gabillon, editors, *SWS*, pages 53–60, Alexandria, VA, USA, 2006. ACM.
- [3] C. Bettini, S. Jajodia, X. S. Wang, and D. Wijesekera. Provisions and obligations in policy rule management. *J. Network Syst. Manage.*, 11(3), 2003.
- [4] J. Biskup and J. Lopez, editors. *Computer Security - ESORICS 2007, 12th European Symposium On Research In Computer Security, Dresden, Germany, September 24-26, 2007, Proceedings*, volume 4734 of *Lecture Notes in Computer Science*. Springer, 2007.

- [5] D. J. Dougherty, K. Fisler, and S. Krishnamurthi. Obligations and their interaction with programs. In Biskup and Lopez [4], pages 375–389.
- [6] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli. Proposed nist standard for role-based access control. *ACM Trans. Inf. Syst. Secur.*, 4(3):224–274, 2001.
- [7] K. Irwin, T. Yu, and W. H. Winsborough. On the modeling and analysis of obligations. In *CCS '06: Proceedings of the 13th ACM conference on Computer and communications security*, pages 134–143, New York, NY, USA, 2006. ACM Press.
- [8] K. LeFevre, R. Agrawal, V. Ercegovic, R. Ramakrishnan, Y. Xu, and D. J. DeWitt. Limiting disclosure in hippocratic databases. In M. A. Nascimento, M. T. Özsu, D. Kossmann, R. J. Miller, J. A. Blakeley, and K. B. Schiefer, editors, *VLDB*, pages 108–119. Morgan Kaufmann, 2004.
- [9] Q. Ni, E. Bertino, and J. Lobo. An obligation model bridging access control policies and privacy policies. In I. Ray and N. Li, editors, *SACMAT*, pages 133–142. ACM, 2008.
- [10] Q. Ni, D. Lin, E. Bertino, and J. Lobo. Conditional privacy-aware role based access control. In Biskup and Lopez [4], pages 72–89.
- [11] Q. Ni, A. Trombetta, E. Bertino, and J. Lobo. Privacy-aware role based access control. In V. Lotz and B. M. Thuraisingham, editors, *SACMAT*, pages 41–50. ACM, 2007.
- [12] J. Park and R. S. Sandhu. The $ucon_{abc}$ usage control model. *ACM Trans. Inf. Syst. Secur.*, 7(1):128–174, 2004.
- [13] C. S. Powers. Privacy promises, access control, and privacy management. In *ISEC '02: Proceedings of the Third International Symposium on Electronic Commerce*, page 13, Washington, DC, USA, 2002. IEEE Computer Society.
- [14] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, 1996.