UNIVERSITY
OF TRENTO

DIPARTIMENTO DI INGEGNERIA E SCIENZA DELL'INFORMAZIONE

38050 Povo – Trento (Italy), Via Sommarive 14
http://www.disi.unitn.it

ONTOLOGY DRIVEN COMMUNITY ACCESS CONTROL

Fausto Giunchiglia, Rui Zhang, and Bruno Crispo

December 2008

Technical Report # DISI-08-080

Also: a short version accepted at the workshop of Trust and Privacy on the Social and Semantic Web (SPOT2009) co-located with the European Semantic Web Conference (ESWC), Crete, 2009

Ontology Driven Community Access Control*

Fausto Giunchiglia, Rui Zhang, and Bruno Crispo

Department of Information Engineering and Computer Science, University of Trento
Via Sommarive 14, Povo 38050 Trento, Italy
{fausto,zhang,crispo}@disi.unitn.it

Abstract. In this paper we present *RelBAC*(for Relation Based Access Control), a model and a logic for access control which models communities, possibly nested, and resources, possibly organized inside complex file systems, as lightweight ontologies, and permissions as relations between subjects and objects. *RelBAC*allows us to represent expressive access control rules beyond the current state of the art, and to deal with the strong dynamics of subjects, objects and permissions which arise in Web 2.0 applications (e.g. social networks). Finally, as shown in the paper, using *RelBAC*, it becomes possible to reason about access control policies and, in particular to compute candidate permissions by matching subject ontologies (representing their interests) with resource ontologies (describing their characteristics).

1 Introduction

The Web 2.0 is making everything happening in the Web more interactive, more social, more dynamic. In turn, this radically changes the scenario within which most applications operate. Among many others, one such scenario and set of applications, taken as reference in this paper, is eBusiness. Internet business patterns such as B2B, B2C, C2C are no longer high-tech terminology but, rather, they represent everyday activities involving virtually everybody from producers to end customers. Businesses exchange information in addition to products via B2B networks; they sell products to customers via B2C networks and customers can even sell their own stuff to one another through C2C interaction patterns. Furthermore, customers are now able to provide feedback for quality and service; sale managers of large companies can distribute advertisements about new products or special offers to the vendors; service companies are able to publish new services through these online media; and so on. Thanks to the Web 2.0, eBusiness can enrich the traditional vending pattern with more active involvement of the involved actors.

However, Web 2.0 applications present new challenges for access control that can be exemplified, taking the eBusiness scenario as a reference, as follows:

* A short version accepted at the workshop of Trust and Privacy on the Social and Semantic Web (SPOT2009) co-located with the European Semantic Web Conference (ESWC), Crete, 2009

- The scale of the business varies greatly from a small personal online shop to the large eBusiness solutions such as Dell. Thus directories of goods could be as simple as several items, or as complex as different product lines including laptop, desktop, printer, etc. What's more, online comments and publishing areas should also be regarded as objects such that different users are allowed to have different flexible fine-grained access. As a consequence, the access control system must be capable of protecting various kinds of objects in largely different scales, possibly organized in complex directory structures.
- The social networks of, e.g., vendors and customers, form an evolving, highly dynamic, soft organization which is usually quite different and independent of the, rather static, enterprise organization. New groups, e.g., new friends or groups associated to a new product such that the iPod, will appear while others will disappear, being obsolete. Permissions, access control rules, policies should be defined in a way which is relatively independent from these evolving structures so that the evolution of the social network has minimal impact on access control policies. This in order to decrease maintenance costs, which are notoriously a key issue in access control.
- The management complexity increases exponentially with the growth of the social structure and shop directories, which in turn, in case of success, tend to expand. The manually created and managed access control rules are time-consuming and error-prone. The emergence of new users and new objects makes the situation far worse as policies must be re-evaluated with any update. There is a need for efficient tools for rule management which would allow to check various properties, such as consistency or separation of duty and to (semi-)automatically generate candidate permissions and access control rules.

RelBAC(for Relation Based Access Control) is a new model and a logic which has been introduced in [1] with the overall goal of dealing with the problem on access control in Web 2.0 applications and which allows to deal with the issues described above. The first key feature of *RelBAC*is that its access control models can be designed using entity-relationship (ER) diagrams. As such, they can be seamlessly integrated into the whole system and vary according to the scale of the business. The second feature, which motivates the name *RelBAC*, is that permissions can be modeled as relations and, as such, and differently from the state of the art, e.g., *RBAC* [2], they can be manipulated as independent objects, thus achieving the requirements of modularity and flexibility described above.

In this paper we take a step further and show how, using *RelBAC*, social networks and object organizations can be modeled as lightweight ontologies (as defined in [3]), by exploiting the translation from classifications and Web directories to Lightweight ontologies described in [4, 5]. This in turn allows us to model permissions as Description Logic (DL) roles [6], access control rules as DL formulas, and policies as sets of DL formulas and, therefore, to reason about access control simply by using off-the shelf DL reasoners, thus addressing the last requirement described above. Some examples of useful *RelBAC*reasoning

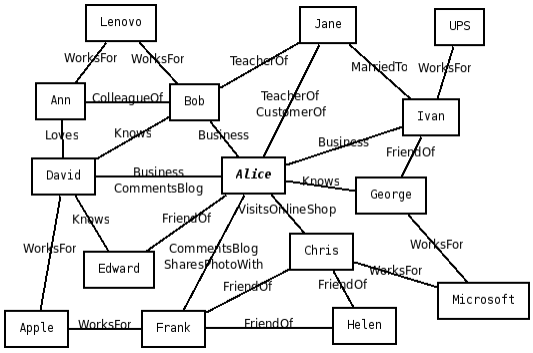

Fig. 1. Alice's Social Network

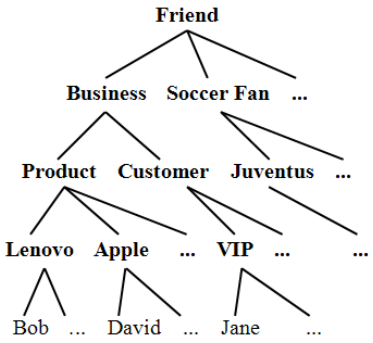

Fig. 2. Alice's Social Ontology

tasks are: whether a user can access a certain resource, the propagation of access rights, the holding of dynamic and static separation of duty, the consistency of a set of policies, and so on. As an important special case of reasoning, in *RelBAC*it becomes also possible to compute candidate permissions by matching user ontologies (representing their interests) with resource ontologies (describing their characteristics) [7–10].

The paper is organized as follows. In Section 2 we introduce the *RelBAC*model and logic. In Section 3 we describe how to implement and manage access control with communities, resources and permissions by representing them, via *RelBAC*, as lightweight ontologies and relations among them. In Section 4 we show how it is possible to reason about access control policies and, in particular, to generate automatically suggestions for permissions by matching subject and object ontologies. Finally, Section 5 describes the related work while Section 6 draws some conclusions.

2 *RelBAC*: Relation Based Access Control

Suppose that Alice, an eBusiness vendor, has an online shop on eBay selling digital devices. Figure 1 shows part of her social network. Thus for instance Bob, David and Ivan have business relations with her, while Chris and George are just common friends. With the continuous growth of this network, Alice decides to manage these contacts in her own way, so that she can easily find the 'proper profiles' whenever necessary. For example, David is a business friend who works at the sales department of Apple, and he will inform Alice about products and special offers such that Alice can immediately put them on her website. Jane is her best customer: she visits Alice's online shop frequently and comments on the deals she has just completed. This will help new potential customers in getting an impression of the service and quality of the goods. And of course, Alice is happy to give Jane VIP prices as rewards. As a consequence, Alice builds the simple tree-like lightweight ontology depicted in Figure 2 to capture, manage and help navigating the messy network of relations of Figure 1.

Fig. 3. The ER Diagram of the *RelBAC* Model.

In this scenario we can see the existence of a quite complex relation structure with a superimposed ontological organization. Notice that the network of relations and the ontology will evolve largely independently. Let us see how we can capture a scenario like this in *RelBAC*.

2.1 The model

We represent the *RelBAC* model as the ER Diagram in Figure 3. Notice that this model is a refined and, at the same time, simpler version of the model presented in [1]. The model has the following components:

- `SUBJECT` (or `USER`): it is a set of subjects (agents in Alice's view) that intend to access some resources. The loop on `SUBJECT` represents the 'IS-A' relation between sets of subjects. The largest subject set is the collection of all the possible subjects.
- `OBJECT`: it is a set of objects or resources that subjects intend to access. The loop on `OBJECT` represents again an 'IS-A' relation between sets of objects. The largest object set is the collection of all the possible objects of the system (e.g., anything with a URI).
- `PERMISSION`: the intuition is that a permission is an operation that subjects can perform on objects. To capture this, a permission is named with the name of the operation it refers to, e.g., *Write* or *Read*. As shown in the ER diagram in Figure 3, a `PERMISSION` is a *relation* between `SUBJECT` and `OBJECT`, namely a set of (subject,object) pairs. The loop on `PERMISSION` represents the 'IS-A' relation between permissions.
- `RULE` (short for `ACCESS CONTROL RULE`): a rule associates a `PERMISSION` to a specific set of `(SUBJECT,OBJECT)` pairs which assigns the specific `SUBJECT` the access right named by the `PERMISSION` onto the specific `OBJECT`. Rules are formalized as DL formulas, as described in the following subsection.

2.2 The Logic

The ER model of *RelBAC*can be directly expressed in DL (see [1] for the details). In general, `SUBJECT`s, and `OBJECT`s are formalized as concepts and single `PERMISSION`s are formalized as DL roles[1]. Individual `SUBJECT`s and `OBJECT`s are formalized as instances and `PERMISSION`s are pairs of instances (`SUBJECT,`

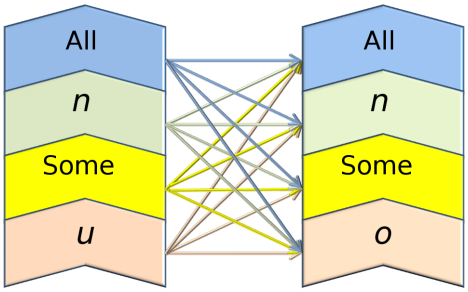

Fig. 4. SUBJECTs and OBJECTs Assignment Combinations

OBJECT). RULEs express the kind of access rights that SUBJECTs have on OBJECTs and are formalized as the *subsumption* axioms provided below.

$U \sqsubseteq \exists P.O$	(1)	$(P : o)(u)$	(7)
$U \sqsubseteq \forall P.O$	(2)	$(\exists P.O)(u)$	(8)
$U \sqsubseteq \geq nP.O$	(3)	$(\forall P.O)(u)$	(9)
$U \sqsubseteq \leq nP.O$	(4)	$(\geq nP.O)(u)$	(10)
$U \sqsubseteq \forall O.P$	(5)	$(\leq nP.O)(u)$	(11)
$U \sqsubseteq P : o$	(6)	$(\forall O.P)(u)$	(12)

In Rule (5) and (12), we abbreviate $\forall \neg P.\neg O$ as $\forall O.P$. This rule allows us to assign a permission P to *all* objects in O. Thus, as schematically represented in Figure 4, we may have a single subject having access to all objects in a set O (rule (12)), to up to/more than n objects (rules (11),(10)), to some objects (rule (8)), to only the objects in O (rule (9)), or to a single object (rule (7)). Dual arguments can be given for any set of users by looking at the rules on the left (rules (1) - (6)). We call the above rules *user-centric*, as they allow us to assign users fine-grained permissions such as those listed above. Dually, we can define corresponding *object-centric* rules by replacing *U, O, P, u, o* respectively with *O, U,* P^{-1}*, o, u*. This feature, not discussed here for lack of space, is however quite important in terms of access control as it allows to define user capabilities.

Thus given a permission P, in *RelBAC*we can write assignments such as the ones described below.

1. the rule 'A user u is allowed to access an object o' is represented as $P(u, o)$. For instance, 'David is allowed to update the entry MB903LL/A' is assigned as $Update(David, MB903LL/A)$;

[1] A DL role is a binary relation, not to be confused with a 'role' of the *RBAC* model.

2. the rule 'A user u is allowed to access some objects in O' is represented as $(\exists P.O)(u)$. For instance, 'David is allowed to update some entries in Digital' is assigned as $(\exists Update.Digital)(David)$;
3. the rule 'A user u is allowed to access maximum n objects in O $(n \leq |O|)$' is represented as $(\leq nP.O)(u)$. For instance 'David is allowed to update at most 5 entries of Digital' is assigned as $(\leq 5Update.Digital)(David)$;
4. the rule 'A user u is allowed to access all objects in O'is represented as $(\forall O.P)(u)$. For instance, 'David is allowed to update all entries in Digital' is assigned as $(\forall Digital.Update)(David)$;
5. the rule 'Some users in U are allowed to access an object o' is represented as $(\exists P^{-1}.U)(o)$. For instance, 'Some friends from Apple are allowed to update the entry MB903LL/A' is assigned as $(\exists Update^{-1}.Apple)(MB903LL/A)$;
6. the rule 'Some users in U are allowed to access some objects in O' is represented as $\exists P.O \sqsubseteq U$. For instance, 'Some friends from Apple are allowed to update some entries in Digital' is assigned as $\exists Update.Digital \sqsubseteq Apple$;
7. the rule 'Minimum m $(m \leq |U|)$ users in U are allowed to access an object o' is represented as $(\geq mP^{-1}.U)(o)$. For instance, 'At least 3 friends from Apple are allowed to update the entry MB903LL/A' is assigned as $(\lg 3Update^{-1}.Apple)(MB903LL/A)$;
8. the rule 'Maximum m users in U are allowed to access some objects in O' is represented as $\leq P^{-1}.U \sqsubseteq O$. For instance, 'At most 3 friends from Apple are allowed to update some entries of Digital' is assigned as $\leq 3Update^{-1}.Apple \sqsubseteq Digital$;
9. the rule 'All users in U are allowed to access an object o' is represented as $U \sqsubseteq P : o$. For instance, 'All friends from Apple are allowed to update the entry MB903LL/A' is assigned as $Apple \sqsubseteq Update : MB903LL/A$;
10. the rule 'All users in U are allowed to access some objects in O' is represents as $U \sqsubseteq \exists P.O$. For instance, 'All friends from Apple are allowed to update some entries of Digital' is assigned as $Apple \sqsubseteq \exists Update.Digital$;
11. the rule 'All users in U are allowed to access all the objects in O' is represented as $U \sqsubseteq \forall O.P$. For instance, 'All friends from Apple are allowed to update all entries of Digital' is assigned as $Apple \sqsubseteq \forall Digital.Update$.

As it can be seen from the above list, *RelBAC*provides a rich set of policies, which becomes even more articulated if one considers object-centric rules. In practice, the most commonly used assignments are the first and the last, which resemble the only two kinds of assignments allowed in $RBAC$.

2.3 The propagation of access rights

Subsumption is used not only for expressing access control `RULE`s. In *RelBAC*, subsumption is also used to represent a partial order $\geq$ among subjects, among objects and among permissions. Figures 2, 5 and 6 provide some examples in the eBusiness scenario. The ordering relation $\geq$ translates the 'IS-A' relation in the *RelBAC*model (see Figure 3) and it allows us to build inheritance hierarchies among subjects, objects and permissions. Inheritance is a very valuable property

as it largely simplifies the otherwise very complex task of administration [2]. We define $\geq$ as follows:

$$U_i \geq U_j \quad iff \quad U_i \sqsubseteq U_j \tag{13}$$
$$O_i \geq O_j \quad iff \quad O_i \sqsubseteq O_j \tag{14}$$
$$P_i \geq P_j \quad iff \quad P_i \sqsubseteq P_j \tag{15}$$

The advantage of having hierarchies on subjects and objects is obvious. The intuition is that smaller sets of subjects and objects are higher in the hierarchy as they correspond to more powerful permissions which in turn will be satisfied by smaller sets of pairs of subjects and objects. The motivation for having hierarchies of permissions is as strong, thought a little more subtle. For example, the permission P_1 '*update the information about some product from a certain IP number during working hours for the purpose of management*' is less powerful than the permission P_2 '*update information about some product*' ($P_2 \geq P_1$). As such, P_1 will be satisfied by more pairs of subjects and objects ($P_2 \sqsubseteq P_1$). In other words, in P_1, with respect to P_2, there will be more subjects which will have more access rights on more objects. Permissions may have rich attributes such as purpose, space, time, condition, etc. and these attributes may make one permission more specific than another. Inheritance hierarchies allow to organize and manage them uniformly rather than one by one, as scattered islands, 'irrelevant' to one another.

3 Lightweight Ontologies for Access Control

Each person will belong to one or more different social communities where many people and different social relations will co-exist. With the communication eased by the development of Internet, social activities such as online forums and blogs greatly increase the number and type of relations in a social network: not only traditional relations like 'knows', 'is-a-friend-of' etc. but new terms such as 'shares-photo-with' or 'comments-on-blog'. On the other hand, people are familiar with tree-like structures such as the file systems of their computer, their email directory, classifications, catalogs, and so on. In general, there is widespread tendency towards organizing resources in tree-like structures. The key feature underlying the success of directories is that one can easily find something according to the property that, the deeper a category is in a tree, the more specific resources it will contain.

Thus access control within user communities can be implemented in *Rel-BAC* as follows:

1. We implement subject hierarchies, as defined Section 2.3, and like the one in Figure 2, as lightweight ontologies (as defined in [3]) of subjects (users, agents, customers, friends, ...);

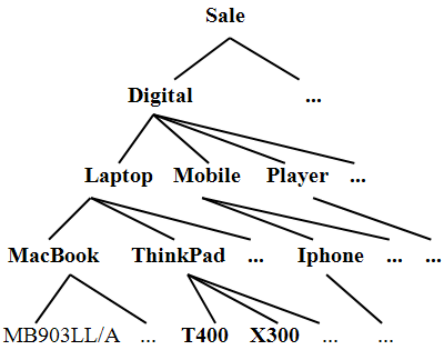

Fig. 5. Alice's Web Directories

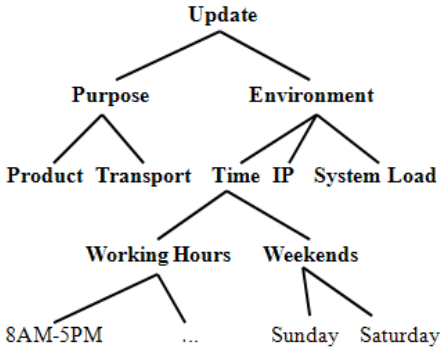

Fig. 6. 'Update' Ontology

2. We implement object hierarchies, as defined Section 2.3, and like the one in Figure 5, as lightweight ontologies of objects (resources, files, photos, web services, ...);
3. We implement permissions as relations and we formalize them as DL roles between subject and object hierarchies. In turn, permissions as well can be organized as lightweight ontologies, like the one in Figure 6, thus implementing the partial order defined in Section 2.3.

However tree-like structures like the ones in Figures 2,5,6, are not quite (lightweight) ontologies, nor does it make sense to propose the use of lightweight ontologies as the main access control mechanism for social networks. The problem is that end users most often do not understand and do not know how to manage structures as sophisticated as (lightweight) ontologies. Our solution is, therefore to translate, with no or very little user intervention, these tree-like knowledge structures into lightweight ontologies. We achieve this goal by exploiting the ideas described in [4, 5]. In these papers, the authors show how a classification or a Web directory can be automatically translated into a lightweight ontology. According to this work, a lightweight ontology is a tree where labels are written in a logical concept language[2] and where each node can be associated a normal form formula which uni-vocally describes its contents. Any classification or directory where each category is labeled with a natural language name expressing its contents, can be translated into a lightweight ontology according to two main steps which can be summarized as follows:

1. The label of each node is transformed into a propositional DL formula using natural language processing (NLP) techniques. For example, the label 'Social Network' is transformed into '$Soccer^i \sqcap Fan^j$' where the superscript $i(j)$ stands for the ith (jth) meaning of the word in a reference dictionary (e.g., WordNet).

[2] A propositional Description Logic without roles.

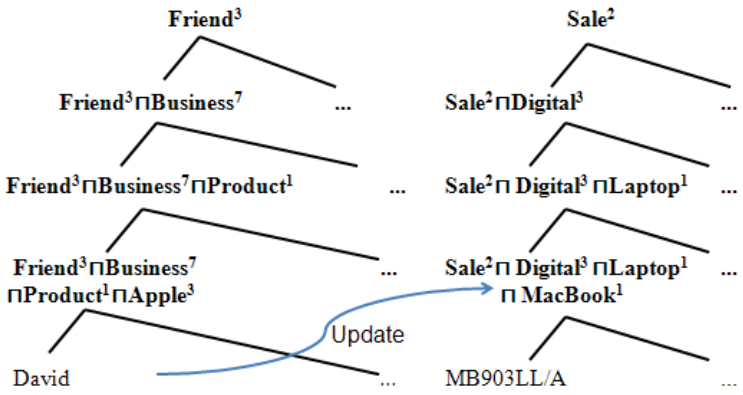

Fig. 7. *RelBAC*Permission Assignment on *LO*s

2. Each node is associated a formula, called the *concept at node*, which is the conjunction of the formulas of all the nodes on the path from the root to the node itself. For example the node labeled 'Soccer Fan' in Figure 2 will be labeled with '$Friend^k \sqcap Soccer^i \sqcap Fan^j$'. The concept at node uni-vocally defines the 'meaning of that node', namely, the set of documents which can be classified under it.

The result of the two steps above is a lightweight ontology where each node is labeled with its concept at node and where each concept at node is subsumed by the concepts of all the nodes above. This property allows for automated document classification and query answering. The user will keep seeing and managing a classification (like the one in Figure 2) but all her operations will be supported and (partially) automated via the background reasoning operating on the underlying lightweight ontology. This background ontology has the same (tree-like) structure as the original classification, but it makes explicit, with its IS-A hierarchy, all the originally implicit and ambiguous relations between object categories. This substantially facilitates the access control problem. More concretely, some advantages are:

- Objects can be automatically classified into the proper directories with the help of a DL reasoner. By exploiting the ideas described in [5] it becomes possible to easily add the vast amount of new information to the proper categories with the proper access rules;
- The evolution of the object ontology (e.g., addition or deletion of a new category) is much more under control because it must satisfy the underlying ontological semantics;
- As from Section 2.3, the permissions on an object category will propagate up the tree.

Considerations similar to those provided for object ontologies apply also to subject ontologies. As mentioned above, these ontologies can be used to organize

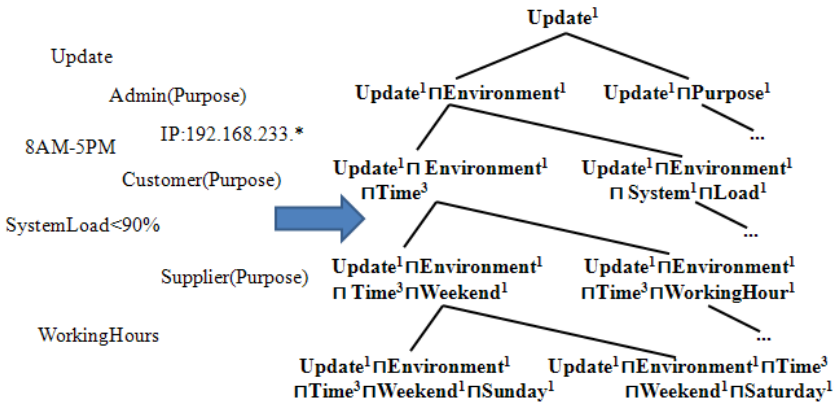

Fig. 8. Scattered Permissions to a *LO*

access to the underlying (possibly very messy) social network (see, for instance Figure 1). There are however two further important considerations. The first is that *RelBAC*subject lightweight ontologies closely resemble *RBAC* role hierarchies [2]. They are however easier to manage as users and permissions are totally decoupled. The second is that the links across subjects in a social network, like those in Figure 1, can be used to suggest candidate paths for permission propagation. One such small example is depicted in Figure 7.

Finally, the translation into a lightweight ontology can be applied also to permission hierarchies. The result of applying this translation process to the hierarchy in Figure 6 is (partially) depicted in the lightweight ontology Figure 8. Notice how the natural language labels in Figure 6 have been translated into DL formulas. The terms on the left of Figure 8 are meant to provide evidence of how the step from natural language to logic allows us to organize otherwise sparse categories. Notice how the lightweight ontology in Figure 8 is upside down with respect the object and subject ontologies presented before. In particular the top category is the most powerful and less populated (in the sense that that it is the one satisfied by the smallest number of subject object pairs). This notation is quite common in access control and it satisfies the intuition that the categories corresponding to the highest number of permissions should be put at the top of the hierarchy.

4 Reasoning about Access Control Rules

The management and administration of access control with complex subject, object and permission structures is quite challenging and error-prone. In *RelBAC*, by exploiting the translation into lightweight ontologies described in the previous section, this activity can be strongly supported by providing tools (i.e., DL reasoners) which automate much of (all?) the reasoning about access control. Some examples of reasoning are:

1. *Design Time Consistency Checking* It's impossible to check by hand a large access control knowledge base, not to say further integration of multiple knowledge bases. Reasoning service of *RelBAC*offers consistency checking such as to check if $\mathcal{S} \cup \mathcal{P} \models \bot$, where $\mathcal{S}, \mathcal{P}$ stand for the knowledge bases corresponding to the state description and rule description. If the answer is negative, the knowledge base is consistent.
2. *Membership Checking* The membership of an individual user or resource to a given set is an important property such as 'Is Bob a member of the business friend group?'. Given the knowledge base with membership declaration such as 'Bob is from Lenovo' and the friend group hierarchy, we can infer $\{Lenovo(Bob), Lenovo \sqsubseteq Product, Product \sqsubseteq Business\} \models Business(Bob)$.
3. *Permission Propagation* An advantage of the hierarchy formalized as 'IS-A' relation through subject, object and permission provides 'free' permission propagation by the reasoning. For example, in the predefined knowledge base we know 'Bob is a business friend', 'write is more powerful than read', 'laptop is a subset of digital device'. Thus we can reason the permission propagation as $\{Bussiness(Bob), Write \sqsubseteq Read, Laptop \sqsubseteq Digital, Business \sqsubseteq \forall Digital.Write\} \models \forall Digital.Write(Bob)$.
4. *Separation of Duty (SoD)* To enforce that some permissions should not be assigned to some users at the same time is the basic idea of *SoD*. For example, 'customers should not be allowed to read and update some category, say Player'. And it's straight forward to be secured by a rule in the knowledge base as $Update : Player \sqcap Read : Player \sqcap Customer \sqsubseteq \bot$. To be precise, "at the same time" can be detailed as design-time and run-time and *SoD* are classified as Static SoD (SSD) and *Dynamic SoD (DSD)*. For example, the previous *SoD* is a kind of *SSD* as it's declared at design time that the two permissions should be separated. If this is allowed in design, but disallowed at run-time, it becomes a *DSD* such as 'customers can be allowed to read and update the Player category, but not physically at the same time'. And this can be reasoned with the *run-time permission* described in [11] as $Updating : Player \sqcap Reading : Player \sqcap Customer \sqsubseteq \bot$
5. *Access Control Decision* At run time, the access control system will face various of access control requests and make decisions at real-time. *RelBAC*turns these requests into formulas and put it to the reasoner and the reasoner will check whether it's consistent with the knowledge base. If 'YES', the request is accepted, otherwise reject.

However the fact that we handle subject, object and permission hierarchies as lightweight ontologies allows us to deal with the problem of semantic heterogeneity, namely with the fact that in general we will have multiple subject and/or object and/ or permission hierarchies which express semantically related notions in many different forms. This problem has been largely studied in the past [7–10]. In the domain of access control this problem becomes quite relevant as being able to deal with that facilitates the management of access control rules with an automated or semi-automated tools for rule creation and reuse. We see two kinds of applications:

1. Two subject hierarchies
2. one subject and one object hierarchy. We found that there exists similarity between the subject and object *LO* although they are heterogeneous ontologies built independently.

Let's go back to Figure 7, it shows parts of the *LO*s built on the hierarchies of Figure 2 and Figure 5. In the left *LO*, David is classified as an instance of the set '$Friend^3 \sqcap Business^7 \sqcap Product^1 \sqcap Apple^3$' according to his social position that he has a $Business^7$ relation with Alice and he works for $Apple^3$ (which is an IT company rather than a fruit). And in the right *LO*, there's a class of objects '$Sale^1 \sqcap Digital^3 \sqcap Laptop^1 \sqcap MacBook^1$' where $Sale^1$ is a branch of $Business^7$, $MacBook^1$ is a $Laptop^1$ as a $product^1$ of $Apple^3$. Apparently the two concepts are different in labels, but semantically overlapping.

To detect these semantic relations between classifications, we use Semantic Matching (S-Match)[7–9]. The original idea is to calculate the semantic similarity such as *equal, overlapping etc.* between the categories of the two given classifications. The core of a S-Match procedure consists two rounds of matching on the *concepts at label* and *concept at node*. The first round can be regarded as a preparation for the second as it discovers the relation between senses from the knowledge base.

WordNet [12] is used as a knowledge base in which possible relations between senses (meanings of word) are provided Semantic similarities are defined with sense relations. *Equal* $\equiv$: one concept is equal to another if there is at least one sense of the first concept, which is a synonym of the second. *Overlapping* $\sqcap$: one concept is overlapped with the other if there are some senses in common. *Mismatch* $\perp$: two concepts are mismatched if they have no sense in common. *More general / specific* $\sqsupseteq, \sqsubseteq$: One concept is more general than the other iff there exists at least one sense of the first concept that has a sense of the other as a hyponym or as a meronym.

For the ontologies of the eBusiness scenario, we can apply the S-Match techniques in two stages: rule creation and rule reuse.

4.1 Suggestions for Rule Creation

For any access control systems, the stage of rules creation is very important because a cute rule set will simplify later work as enforcement and management. Semantic Matching between the subject, object ontologies will clarify all the semantic relations between categories of the two *LOs*. For example, given the background knowledge about the relations $MacBook^1$ is a $Laptop^1$ as a $product^1$ of $Apple^3$ etc., we can find the semantic similarities as list in Table 1. As WordNet does not 'know' the word as 'MacBook', which is common under the enormous birth of new words in this Internet era, we should enrich the knowledge base with the facts such as '*$Apple^3$ is a IT company selling digital products such as MacBook and IPod.*'.

¿From Table 1, we can see the semantic similarities such as $Sale^2 \sqsubseteq Business^7$, etc. The interesting thing here is that the relation between $Friend^3$ and $Sale^2$

Table 1. Semantic Matching on Labels

S-Match	$Friend^3$	$Business^7$	$Product^1$	$Apple^3$	$Lenovo^1$	$Soccer^1 \sqcap Fan^2$
$Sale^2$	$\bot$	$\sqsubseteq$				$\bot$
$Digital^3$						
$Laptop^1$			$\sqsubseteq$			
$MacBook^1$				$\sqcap$	$\bot$	
$Thinkpad^1$				$\bot$	$\sqcap$	

is $\bot$ which means that the two words has nothing in common. It's weird but true as $Friend^3$ means '*a person with whom you are acquainted*' and $Sale^2$ is '*the general activity of selling*'. And this will be the general case when we try to match a subject ontology with an object ontology. If we went on with the second step to match the concepts at node in the LOs, $\bot$s will fill up all the table, which is not usable for rule creation. So, we only use the first round S-Match to discover the relations. These relations provide the following suggestions to create new rules.

Semantically Related The cells marked with '$\sqsubseteq, \sqsupseteq, \equiv, \sqcap$' represent the semantic similarity of the corresponding concepts. And it's meaningful to assign corresponding users some access to the objects. For example, the relation $Sale^2 \sqsubseteq Business^7$ suggests that some access, let's say *Read*, should be assigned to the $Business^7$ $Friend^3$ to some $Sale^2$ categories. It's obvious here in the small toy user and object ontologies, but facing a large eBusiness such as Amazon.com, these similarities will be very useful for the administrators while creating new rules.

Explicit Unrelated The cells marked with '$\bot$[3]' represent that the corresponding concepts are found 'unrelated' in the knowledge base. We have to differentiate the real world semantics of these '$\bot$'s.

- $Sale^2 \bot Friend^3$ is a *Subject/object mismatch* as they are mismatch only because they are referring to person and activity respectively;
- $MacBook^1 \bot Lenovo^1$ comes from that '*MacBook is a product of Apple company but not Lenovo.*' And this explicit mismatch relation between the two concepts suggests no access should be assigned;
- $Sale^2 \bot (Soccer^1 \sqcap Fan^2)$ covers both of the upper cases so it's still explicit unrelated and no access should be assigned.

I don't know (IDN) The blank cells of the table means that the knowledge base doesn't know any existing relation between the corresponding concepts. And it's up to the administrators to decide whether to assign some access or not. From Table 1 we can see that this kind of cells are the majority because the knowledge base we use is not specially for eBusiness domain and if the it is specially enriched, more semantic relations will be found and more suggestions will be offered.

[3] Here we shorten the axiom '$C_1 \sqcap C_2 \sqsubseteq \bot$' as '$C_1 \bot C_2$'.

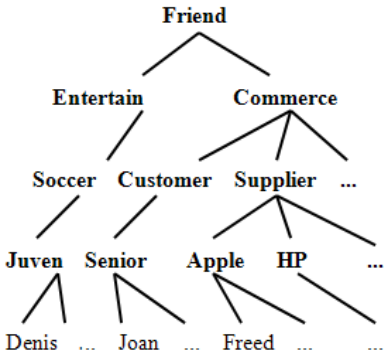

Fig. 9. Bob's Social Ontology

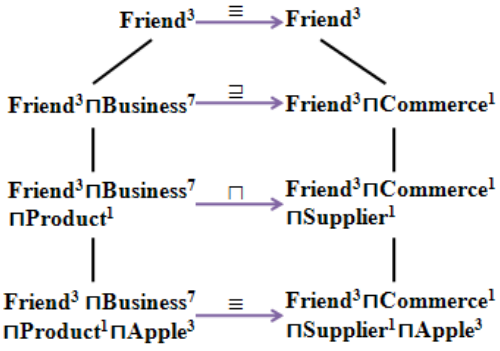

Fig. 10. Ontology Matching for Rule Reuse

4.2 Automated Rule Reuse

One important evolution of subject and object ontologies is to integrate similar ontologies. For example, an eBusiness vendor will enlarge her social network to involve more customers and very likely to integrate the customer ontology of another vendor, or symmetrically to integrate the goods ontology. The traditional access control solutions require the administrator to create new rules for these evolving parts. Even for the similar ontologies, all assignments have to be made once more. For example, the vendor in the scenario of Section 2 would like to merge another ontology of subjects as Bob's Social Ontology as Figure 9. It is also an ontology about friends, but in different structures and descriptions of the person sets. For instance, a customer set called 'Senior' has the similar intuition to the 'VIP' set in previous ontology.

We shown in Figure 10 the results of S-Match on two branches of the 'friend' ontologies in Figure 2 and 9. The semantic similarity axioms can be added to the knowledge base of access control and the rule reuse is done without further efforts. For example,

$$\{(Friend^3 \sqcap Commerce^1) \sqsubseteq (Friend^3 \sqcap Business^7), Business^7 \sqsubseteq \alpha\}$$
$$\models Friend^3 \sqcap Commerce^1 \sqsubseteq \alpha$$

So any *subject-centric* rules that assigned to $Business^7$ permissions will also propagate to $Friend^3 \sqcap Commerce^1$ without creating new rules for the new subject sets. Similar reuse applies on objects as well when S-Match is used to find the semantic similarities of the object ontologies.

5 Related Work

With the arrival of Web 2.0 and now coming even Web 3.0, privacy and access control over the resources online throughout the evolving social networks stretching out with the expanding Internet arouse more and more attention.

RBAC [2], a popular access control model in many fields has been tried to be applied on this new dynamic scenario. Its popularity in enterprise solutions such as Sun and IBM is a proof of its efficiency, but the permission propagation through predefined role hierarchies, as an advanced feature, is no longer suitable or even a drawback for dynamic environment such as the social community and web directories. In contrast, *RelBAC*provide an adjustable model for dynamic subject and object structures. In addition, lightweight ontologies [3, 13, 14] and semantic matching [7–9, 15–17] techniques help a lot in formalizing the dynamics and heterogeneous structures.

Caminati et al. proposed a solution based on cryptographic and digital signature techniques in [18]. This kind of access control systems focus on protection of security threats rather than taking use of the rich information of the social network. And the authentication procedure is done once for all the life cycle of a live session which is not enough for the need of fine grained access control.

Lockr[19] is proposed to fit the situation nowadays that the large number of content sharing systems and sites use different access control methods and not reusable for each other. It separates social networking information from from the content sharing mechanisms, so that end users do not have to maintain several site-specific copies of their social networks. It also provides a way to use social relationships as an important attribute, *relationship type*, to define access control rules. But the core of Lockr is still a public/private key communication which is short of fine-grained rule specification such as we can do with the 15 ways of rules in *RelBAC*.

Another thread similar to our solution is SAC (Semantic Based Access Control). Yague et al. discuss the SAC (Semantic Access Control) model in [20] with a XML based language SPL (Semantic Policy Language). The SAC model is based on the semantic properties of the resources, clients (users), contexts and attribute certificates. And it relies on the rich expressiveness of the attributes to create and validate access control. The flexibility of SAC to define access control over attributes also brings the complexity problem of system as the number of rules explodes with the number of attributes which is far more than the number of resources and clients. In contrast, our model covers the expressiveness of attributes and take use of the structure at the same time so that the permission propagation will greatly reduce the number of rules. Pan et al. present a novel middle-ware based system [21] to use semantics in access control. It is based on *RBAC* model [2] with the core as semantic scopes of objects and concepts. Access control rules are all defined on concepts, which are in turn predefined by attributes. For interoperation purpose, they use semantic mapping on roles in order to find the similarity or separation of duties between roles in two ontologies. And this is similar to our approach, but we do much further as the S-Match tools are not domain specific so that we can match from a subject ontology to an object ontology, result of which offers suggestions for rule creation.

6 Conclusion

In this paper we have presented *RelBAC*, a new model and logic for access control. The main feature of *RelBAC*is that it allows to organize users and objects as (lightweight) ontologies and that it models permissions are relations. This in turn allows to represent access control rules and policies as DL formulas and therefore to reason about them using state of the art off-the-shelf reasoners. In turn, as shown in the second part of the paper, this allows us to match, using the semantic matching technology, the user and the object ontologies and, as a consequence, to (semi)- automatically permissions which (may) fit the user interests. The idea is that these permissions are then proposed to the user as suggestions to be confirmed and approved.

References

1. Giunchiglia, F., Zhang, R., Crispo, B.: Relbac: Relation based access control. In Society, I.C., ed.: International Conference on Semantics, Knowledge and Grid, SKG 2008. (2008)
2. Ferraiolo, D.F., Sandhu, R.S., Gavrila, S.I., Kuhn, D.R., Chandramouli, R.: Proposed NIST standard for role-based access control. Information and System Security **4**(3) (2001) 224–274
3. Giunchiglia, F., Zaihrayeu, I.: Lightweight ontologies. In: Encyclopedia of Database Systems, Springer (2008)
4. Giunchiglia, F., Marchese, M., Zaihrayeu, I.: Encoding classifications into lightweight ontologies. In: ESWC. (2006) 80–94
5. Giunchiglia, F., Marchese, M., Zaihrayeu, I.: Towards a theory of formal classification. In: CandO 2005,AAAI-05, Pittsburgh, Pennsylvania, USA (July 9-13 2005 2005)
6. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F., eds.: The description logic handbook: theory, implementation, and applications. Cambridge University Press, New York, NY, USA (2003)
7. Giunchiglia, F., Yatskevich, M., Shvaiko, P.: Semantic matching: Algorithms and implementation. Journal on Data Semantics (2007) 1–38
8. Giunchiglia, F., Yatskevich, M., Giunchiglia, E.: Efficient semantic matching. In: Proceedings of the 2nd European semantic web conference (ESWC'05), LNCS, Springer Verlag (2005)
9. Giunchiglia, F., Yatskevich, M.: Element level semantic matching. In: Meaning Coordination and Negotiation workshop at ISWC'04, Hiroshima, Japan (November 2004)
10. Giunchiglia, F., Shvaiko, P., Yatskevich, M.: Semantic schema matching. In: In Proceedings of CoopIS, Springer (2005) 347–365
11. Zhang, R., Crispo, B., Giunchiglia, F.: Relbac: Design and run-time reasoning about web access control policies. Technical report, University of Trento (2008)
12. Miller, G.A.: Wordnet: A lexical database for english. Communications of the ACM **38** (1995) 39–41
13. Giunchiglia, F., Marchese, M., Zaihrayeu, I.: Encoding classifications into lightweight ontologies. J. Data Semantics **8** (2007) 57–81

14. Zaihrayeu, I., Sun, L., Giunchiglia, F., Pan, W., Ju, Q., Chi, M., Huang, X.: From web directories to ontologies: Natural language processing challenges. In: In 6th International Semantic Web Conference (ISWC, Springer (2007)
15. Giunchiglia, F., Shvaiko, P., Yatskevich, M.: Discovering missing background knowledge in ontology matching. In: ECAI. (2006) 382–386
16. Giunchiglia, F., Shvaiko, P., Yatskevich, M.: Semantic schema matching. In: In Proceedings of CoopIS, Springer (2005) 347–365
17. Giunchiglia, F., Yatskevich, M., Shvaiko, P.: Semantic matching: Algorithms and implementation. J. Data Semantics **9** (2007) 1–38
18. Carminati, B., Ferrari, E.: Privacy-aware collaborative access control in web-based social networks. In Atluri, V., ed.: DBSec. Volume 5094 of Lecture Notes in Computer Science., Springer (2008) 81–96
19. Tootoonchian, A., Gollu, K.K., Saroiu, S., Ganjali, Y., Wolman, A.: Lockr: social access control for web 2.0. In: WOSP '08: Proceedings of the first workshop on Online social networks, New York, NY, USA, ACM (2008) 43–48
20. del Valle, M.I.Y., del Mar Gallardo, M., Mana, A.: Semantic access control model: A formal specification. In di Vimercati, S.D.C., Syverson, P.F., Gollmann, D., eds.: ESORICS. Volume 3679 of Lecture Notes in Computer Science., Springer (2005) 24–43
21. Pan, C.C., Mitra, P., Liu, P.: Semantic access control for information interoperation. In: SACMAT '06: Proceedings of the eleventh ACM symposium on Access control models and technologies, New York, NY, USA, ACM (2006) 237–246