Ontology Driven Community Access Control*

Fausto Giunchiglia, Rui Zhang, and Bruno Crispo

Dipartimento di Ingegneria e Scienza dell'Informazione , 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, social and 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 terminologies 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 feedbacks for quality and service; sales 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 longer version of this work can be found as a DISI technical report at http://eprints.biblio.unitn.it/archive/00001527/01/080.pdf

- The scales of eBusinesses may differ greatly from small personal online shops to large eBusiness solutions such as Dell. Thus, directories of goods could be as simple as several items, or as complex as multiple product lines including laptop, desktop, printer, etc. 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. Permissions, access control rules and policies should be defined relatively independently so that the evolution of the social network has minimal impact on access control policies.
- 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. Manual rule creation and management are time-consuming and error-prone. Efficient tools for rule management are crucially necessary which would allow to check various properties, such as consistency or separation of duties 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. 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 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]. This in turn allows us to model permissions as Description Logic (DL) roles [5], 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.

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.

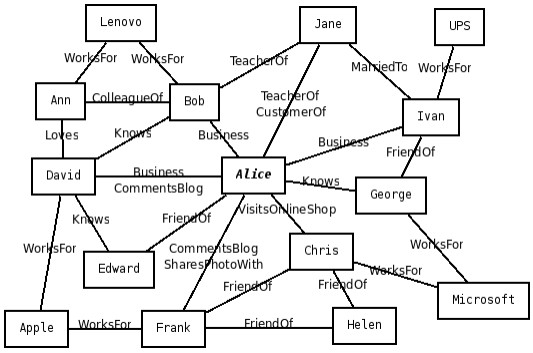

Fig. 1. Alice's Social Network

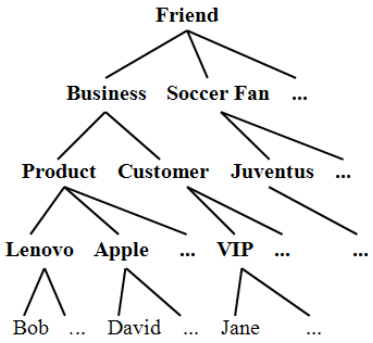

Fig. 2. Alice's Social Ontology

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 has 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 potential customers to get an impression of the service and quality of the goods. Therefore, 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 Figure 1. Let us see how we can capture a scenario like this with *RelBAC* in this section.

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).

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

- `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. In general, `SUBJECT`s, and `OBJECT`s are formalized as concepts and `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 i.e. (`SUBJECT, OBJECT`). `RULE`s express the kind of access rights that `SUBJECT`s have on `OBJECT`s and are formalized as the *subsumption* axioms provided below. In Rules 6 and 12, we abbreviate $\forall\neg P.\neg O$ as $\forall O.P$, which allows us to assign a permission P to *all* objects in O. Thus, we may have a single subject 'u' having access to a single object (Rule 7), to some objects (Rule 8), to only the objects in O (Rule 9), to minimum or maximum n objects (Rules 10 and 11), or to all objects in a set O (Rule 12). Dual arguments can be given for any set of users 'U' by looking at the rules on the left (Rule 1 - 6). We call these 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 design policies from different perspectives.

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

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

Thus given a permission P, in $RelBAC$ we can write permission assignment policies such as the ones described below.

1. 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 $Apple \sqsubseteq Update : MB903LL/A$;
2. The rule 'all users in U are allowed to access some objects in O' is represented as $U \sqsubseteq \exists P.O$. For instance, 'all friends from Apple are allowed to update some entries of Digital' is assigned as $Apple \sqsubseteq \exists Update.Digital$;
3. The rule 'all users in U are allowed to access minimum (maximum) n objects in O' is represented as $U \sqsubseteq \geq (\leq) nP.O$. For instance, 'all friends from Apple are allowed to update minimum (maximum) 5 entries of Digital' is assigned as $Apple \sqsubseteq \geq (\leq) 5Update.Digital$;
4. The rule 'all users in U are allowed to access only the objects in O' is represented as $U \sqsubseteq \forall P.O$. For instance, 'all friends from Apple are allowed to update only entries of Digital' is assigned as $Apple \sqsubseteq \forall Update.Digital$.
5. 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

In $RelBAC$, subsumption is not only used to express access control `RULE`s but also used to represent the partial order '$\geq$' among subjects, among objects and among permissions. 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 address 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. These attributes may make one permission more specific (less powerful therefore more assigned pairs) 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

With the communication simplified 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'. In another perspective, people are familiar with tree-like structures such as the file systems of their computers, their email directories, classifications, catalogs, and so on. In general, there is a widespread tendency towards organizing resources in tree-like structures. The key feature underlying the success of tree-like 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, community access control can be implemented in *RelBAC* with the subjects, objects and permissions encoded into different 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], in which the authors show how a classification or a Web directory can be automatically translated into a lightweight ontology. 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, 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 'Soccer Fan' 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. 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 univocally 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 object

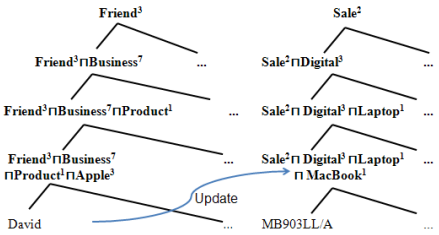

Fig. 4. Permission Assignment on Lightweight Ontologies

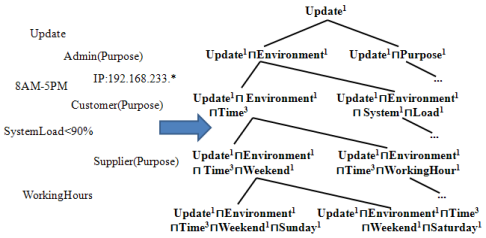

Fig. 5. Scattered Permissions to a Lightweight Ontology

classification and query answering. People will keep seeing and managing a classification (like the one in Figure 2) but all their 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 contributes to address 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 [4] 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 category) is much more under control because it must satisfy the underlying ontological semantics;
- With the partial order formalized as in Section 2.3, the permissions on an object category will propagate up the tree without extra policies (discussed more in Section 4).

Considerations similar to those provided for object ontologies apply also to subject ontologies. As mentioned above, these ontologies can be used to organize 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 4.

Finally, the translation into a lightweight ontology can be applied also to permission hierarchies. Notice that natural language labels have been translated into DL formulas. The terms on the left of Figure 5 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 5 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 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 are quite challenging and error-prone. In $RelBAC$, by exploiting the translation into lightweight ontologies described in Section 3, these activities can be strongly supported by providing tools (i.e., DL reasoners) which automate much (if not all) of the reasoning about access control such as design time ontology consistency checking, permission propagation management, separation of duties, etc. Some examples of reasoning are:

Design Time Consistency Checking It is almost impossible to check manually a large access control knowledge base, not to say further integration of multiple knowledge bases. The 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 policy description. If the answer is negative, the knowledge base is consistent.

Permission Propagation An advantage of the hierarchy formalized as 'IS-A' relations through subjects, objects and permissions provide '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 $\{Bussiness(Bob), Write \sqsubseteq Read, Laptop \sqsubseteq Digital, Business \sqsubseteq \forall Digital.Write\} \models (\forall Digital.Write)(Bob)$.

Separation of Duties (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* as $(Updating : Player) \sqcap (Reading : Player) \sqcap Customer \sqsubseteq \bot$.

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 a request into a formula and put it to the reasoner and then the reasoner will check whether it is consistent with the knowledge base. A positive answer means that the request is acceptable, otherwise should be denied.

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 addressed as *semantic matching* in [6]. In the domain of access control this problem becomes quite relevant as we see two kinds of applications of the semantic matching techniques.

1. Two hierarchies of the same kind such as two subject hierarchies, two object permission hierarchies, etc.
2. One subject and one object hierarchy. We found that there exists similarity between the subject and object lightweight ontology although they are heterogeneous ontologies built independently.

Let's go back to Figure 4, it shows parts of the lightweight ontologies built on two hierarchies, one subject and one object. On the left, 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). On the right, there's a class of objects '$Sale^2 \sqcap Digital^3 \sqcap Laptop^1 \sqcap MacBook^1$' where $Sale^2$ 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 semantic relations between lightweight ontologies, we use S-Match as described in [6]. The original idea is to calculate the semantic similarity such as *equal, overlapping*, etc. between the categories of the two given classifications. For the ontologies of the eBusiness scenario, we can apply the S-Match techniques in two stages: rule creation and rule reuse. Let us consider them in turn.

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 lightweight ontologies. For example, given the background knowledge about the relations such as '$MacBook^1$ is a $Laptop^1$ as a $Product^1$ of $Apple^3$', we can find the semantic similarities as listed in Table 1. These relations may provide suggestions as follows.

Semantically Related The cells marked with '$\sqsubseteq, \sqsupseteq, \equiv, \sqcap$' represent the semantic similarity (*less general, more general, equal, overlapping*) of the corresponding concepts. It is rational to assign some access to users over the objects that are semantically related. For example, as $Sale^2$ is semantically subsumed by $Business^7$, most probably the subjects as instances of $Business^7$ should have some access over objects as instances of $Sale^2$.

Table 1. Semantic Matching on Labels

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

Explicit Unrelated The cells marked with '$\perp$' represent that the corresponding concepts are found 'unrelated' in the knowledge base. Here we shorten the axiom '$C_1 \sqcap C_2 \sqsubseteq \perp$' as '$C_1 \perp C_2$'. We have to differentiate the real world semantics of these '$\perp$'s.

- $Sale^2 \perp Friend^3$ is a mismatch because they are referring to object and subject, i.e. an activity and a person respectively. This mismatch comes from the disjointness between person and activity as different subjects but does not prevent that a person can have some relation with an activity such as $Friend^3$ may have access to $Sale^2$.
- $MacBook^1 \perp Lenovo^1$ comes from that '*MacBook is a product of Apple company but not Lenovo.*' This kind of mismatch suggests exactly no access should be assigned.
- $Sale^2 \perp (Soccer^1 \sqcap Fan^2)$ covers both upper cases so it does prevent the access assignment from $Soccer^1$ Fan^2 to $Sale^2$.

I don't know The blank cells of the table mean that the knowledge base doesn't know any existing relations between the corresponding concepts. These cases are left to the administrators to decide whether to assign some access or not.

4.2 Automated Rule Reuse

One important evolution of subject and object ontologies is to integrate similar ontologies. For example, eBusiness vendors would like to enlarge their social networks to involve more customers and very likely to integrate customer ontologies of other vendors, or symmetrically to integrate the goods ontology. In classical access control solutions, administrators have to create new rules for these evolving parts. Even for similar ontologies, all assignments have to be made once again. For example, Alice, the eBusiness vendor in the scenario of Section 2 would like to merge another ontology of subjects, say, Bob's Social Ontology partly shown as Figure 6. 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 have shown in Figure 7 the results of S-Match on two branches of the 'friend' ontologies in Figures 2 and 6. 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,

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

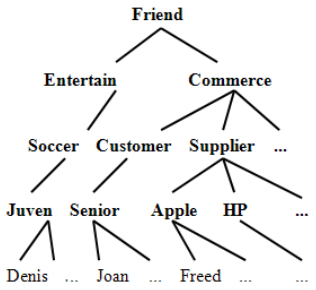

Fig. 6. Bob's Social Ontology

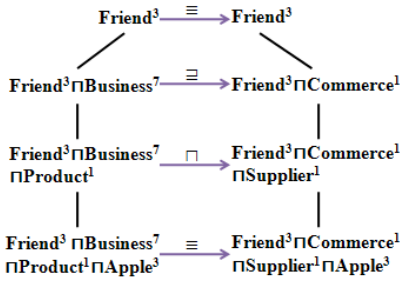

Fig. 7. Ontology Matching for Rule Reuse

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

5 Related Work

Classic access control techniques, e.g., cryptography have been proposed for community access control such as [7]. However, this kind of access control systems focus on protection from security threats rather than taking use of the rich information from the web. Lockr[8] was proposed to fit the situation that the large number of content sharing systems and sites use different access control methods un-reusable for each other. It separates social networking information 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. However, Lockr still uses a public/private key communication and does not consider the semantic similarities.

Another series of research focus on providing policy languages for the rich semantics on the web such as KAoS[10], Rei[11], etc. Yague et al. in [9] even presented a model named Semantic Based Access Control. The model is based on the semantic properties of the resources, clients (users), contexts and attribute certificates and relies on the rich expressiveness of the attributes to create and validate access control policies.

Pan et al. present a novel middle-ware based system [12] to use semantics in access control based on the $RBAC$ model [2] with a mediator to translate the access request between organizations by replacing roles and objects with matched roles and matched objects. They used semantic mapping on roles in order to find the similarity or separation of duties between roles in two ontologies. We do further as the S-Match tools are more generic and can match a subject ontology with an object ontology in order to suggest new rules.

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 generate (semi-)automatically permissions which (may) fit the user interests. The idea is that these permissions are then proposed to the administrator 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. Number 978-0-387-35544-3. In: Encyclopedia of Database Systems. Verlag, Springer (June 2009)
4. Giunchiglia, F., Marchese, M., Zaihrayeu, I.: Towards a theory of formal classification. In: CandO 2005,AAAI-05, Pittsburgh, Pennsylvania, USA (2005)
5. 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)
6. Giunchiglia, F., Yatskevich, M., Shvaiko, P.: Semantic matching: Algorithms and implementation. J. Data Semantics **9** (2007) 1–38
7. 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
8. 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
9. 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 LNCS., Springer (2005) 24–43
10. Uszok, A., Bradshaw, J., Jeffers, R., Suri, N., Hayes, P., Breedy, M., Bunch, L., Johnson, M., Kulkarni, S., Lott, J.: Kaos policy and domain services: Toward a description-logic approach to policy representation, deconfliction, and enforcement. Policies for Distributed Systems and Networks, IEEE International Workshop on **0** (2003) 93
11. Kagal, L.: Rei: A policy language for the me-centric project. Technical report (2002)
12. 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