Design and Application of Rule Based Access Control Policies

Huiying Li, Xiang Zhang, Honghan Wu, Yuzhong Qu

Department of Computer Science and Engineering, Southeast University,
Nanjing 210096, P.R.China
{huiyingli, xzhang, hhwu, yzqu} @seu.edu.cn

Abstract. Access control is an important issue among the security problems of resources in distributed systems. In order to enable entities in distributed systems to understand and interpret policies correctly, common concern is drawn to the problem of expressing access control policies with semantic information. In this paper, we introduce how to express access control policies based on OWL and SWRL. It includes a definition of OWL ontology to describe the terms and a declaration of SWRL rules to explicit the relationship between properties. Finally some use cases are given to explain how to express policies in form of rule using the terms defined beforehand.

1 Introduction

The dissemination and manipulation of information resources across large-scale networks of computers is increasingly prevalent. As a result, common concern is drawn to the security of resources in distributed systems, which deals with many aspects, such as identification and authentication of users, encryption of resources and access control issue. Instead of dealing with all the aspects, this paper introduces how to design rule-based access control policies using semantic language-OWL (Web Ontology Language)[10] and rule language-SWRL (Semantic Web Rule Language)[3].

Policy language has been studied for a long time, there are also some languages can express access control policy, such as XACML (eXtensible Access Control Markup Language)[12], X-RBAC (XML Role-Based Access Control)[6], KAoS(Knowledge-able Agent-oriented System)[5], Rei[8] and Ponder[9]. Among them, XACML is an OASIS standard specification now. It defines a general policy language based on XML used to protect resources as well as an access decision language. X-RBAC is based on an extension of the role-based access control model, it provides a framework for specifying mediation policies in a multidomain environment and allows specification of RBAC policies and facilitates specification of timing constraints on roles and access requirements as well [6]. Both XACML and X-RBAC are based on XML, and they have had broad

applications in some enterprise environments. However it is difficult for them to deal with the inter-operation at the level of semantics, because the entities and relationships defined by XML are lack of formal meaning. In order to enable entities in distributed systems to understand and interpret polices correctly, it will be better to represent policy language in semantic language such as RDF-S, DAML+OIL or OWL[7]. KAoS and Rei both have accomplished some work in this regard. Providing an open distributed architecture for software agents is the initial purpose to develop KAoS[5], it could also express the agents action control policy relying on the DAML-based ontology in Web/Grid Service environment. Represented in OWL-Lite, Rei is a policy language based on deontic concepts and includes constructs for rights, prohibitions, obligations and dispensations [8]. Another policy language is Ponder, which is a declarative, object-oriented language that can be used to specify both security and management policies [9]. Different from the policy languages discussed above, we design the access control policies based on OWL and SWRL in the form of rules using entities defined in advance.

As mentioned above, policy language expressed in ontology language has the advantage of dealing with inter-operation at semantics level. We define ontology to help express access control policies using OWL, which added considerable expressive ability. But OWL still has the expressive limitations, particularly with respect to what can be said about properties [1]. For enhancing the expressive and deducible ability, we also define some rules to explain the relationship between properties using SWRL, a Horn clause rules extension to OWL.

The paper is structured as follows: a brief introduction to SWRL is given in Section 2. Following this, the ontology and rules designed by us are presented in Section 3. In Section 4, we talk about the advantage of our method to express policy using some use cases. Finally, the summary of our work and future research directions are discussed in Section 5.

2 Semantic Web Rule Language

SWRL is a Horn clause rules extension to OWL proposed as a W3C member submission in 2004. It extends OWL DL by adding a simple form of Horn-style rules in a syntactically and semantically coherent manner for the purpose of enhancing expressive ability.

The main axiom added to OWL DL is Horn clause rules, which are of the form of an implication between an antecedent (body) and consequent (head). The informal meaning of a rule can be read as: whenever (and however) the conditions specified in the antecedent hold, then the conditions specified in the consequent must also hold.

A rule written in an informal human readable syntax has the form below:

$$\text{antecedent} \rightarrow \text{consequent}.$$

Both the antecedent and consequent of a rule consist of zero or more atoms. Atoms can be of the form C(x), P(x, y), Q(x, z), sameAs(x, y) or differentFrom(x, y), where C is an

OWL DL description which may be a class name or a complex description using Boolean combination, restrictions etc. P is an OWL DL individual-valued Property, Q is an OWL DL data-valued Property, and x, y are either variables or OWL individuals, z is either a variable or an OWL data value. Informally, an atom C(x) holds if x is an instance of the class description C, an atom P(x, y) (or Q(x, z)) holds if x is related to y(z) by property P(Q), an atom sameAs(x, y) holds if x is interpreted as the same object as y, and an atom differentFrom(x, y) holds if x and y are interpreted as different objects.

Having the rules discussed above, it will be easy to assert more complex relationships between properties, such as the composition of two properties. A typical example is showed as follows:

$$\text{parent}(?a, ?b) \wedge \text{brother}(?b, ?c) \rightarrow \text{uncle}(?a, ?c).$$

Hereinto, variables are indicated using the standard convention of prefixing them with a question mark, parent and brother are both OWL properties. Then the rule above asserts that the composition of parent and brother properties implies the uncle property. In other words, if John has Mary as a parent and Mary has Bill as a brother, then this rule requires that John has Bill as an uncle.

Using SWRL, we define some rules similar with the one presented above to express relationships between the properties in our OWL ontology. It makes the meaning of the property more clear. We also allow users to assert their own access control policies in the form of rules using entities defined in our ontology.

3 Design of Rule Based Access Control Policies

In overview, we design access control policies in the form of rules using entities defined in our OWL ontology. The ontology includes concepts about agent and resource and properties between them such as **isPermittedDoWith** and **isProhibittedDoWith**. We also express some rules using SWRL to describe the relationships between the properties defined in the ontology. Currently, our application domain is the project of WonderSpace, which is a Semantic Web application developed by our Lab (http://xobjects.seu.edu.cn/). Its potential users are researcher groups. The project aims at speeding up their research and study, and bringing high efficiency in collaborations by providing an environment for resource and knowledge sharing.

3.1 The ontology

The purpose of designing our ontology is to help users express their access control policies in WonderSpace, the core classes and properties are defined as Fig. 1.

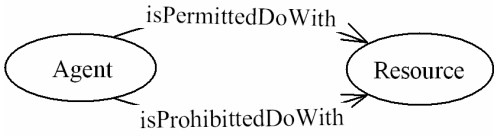

Fig. 1. Graph model of the core classes

Using this model, user can assert that what kinds of agents are permitted/prohibited to access to what kinds of resources. The access control policy is associated with the properties of the agent instead of identities. For example, a research group leader published a resource and specified the access control policy like this: only PhD students in my group are permitted to download this resource. Such policy can be expressed in the form of rules using classes and properties defined in our ontology. For the reason of providing more properties for user to describe agent and resource, we design the Agent class and Resource class showed in Fig. 2 and Fig.3. We also specify many subproperties of **isPermittedDoWith** and **isProhibittedDoWith** to express more actions that can/cannot take to the resource, such as **isPermittedPublishWith**, **isPermittedDeleteWith**, **isPermittedDownloadWith**, **isPermittedReadWith**, **isPermittedUpdateWith**, which are all the subproperties of **isPermittedDoWith**, while **isProhibittedPublishWith**, **isProhibittedDeleteWith**, **isProhibittedDownloadWith**, **isProhibittedReadWith**, **isProhibittedUpdateWith** are all subproperties of **isProhibittedDoWith**.

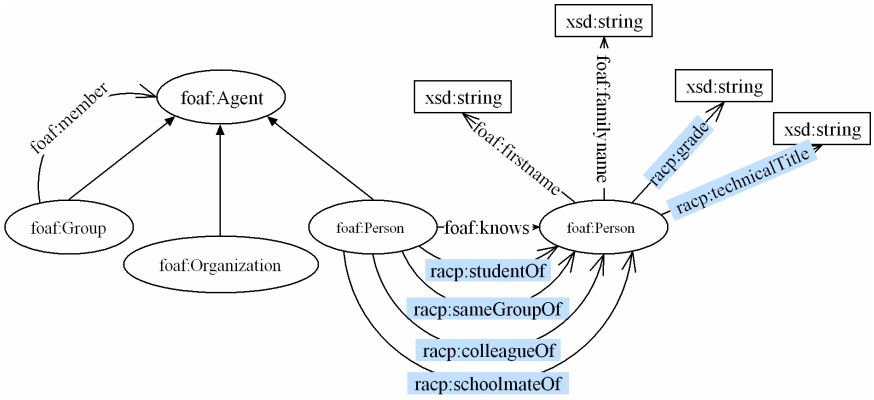

Fig. 2. Agent ontology

Fig. 2 shows the main properties and subclasses of Agent class, the entities are borrowed from the ontology of Friend of a Friend (FOAF) project. FOAF project is about creating a Web of machine-readable homepages describing people, the links between them and the things they create and do (http://www.foaf-project.org/). FOAF has already done a good work about developing ontology about Agent, so we use the basic classes and

properties in the FOAF ontology for reference. For the purpose of expressing domain information about agent, we also extend the ontology by adding some properties, some of them are DatatypeProperty such as **grade**, **technicalTitle**, some are ObjectProperty to describe the relationship between peoples such as **studentOf**, **sameGroupOf, colleagueOf**, **schoolmateOf**. We define these four objectproperties as the subproperty of **knows**, and some of them had subproperties likewise, for example, the **phdStudentOf**, **masterStudentOf**, **collegeStudentOf** are the subproperty of **studentOf.**

The resource ontology showed in Fig. 3 is the one used in the WonderSpace now. It denotes the classification of resources and figures out some attributes of resources.

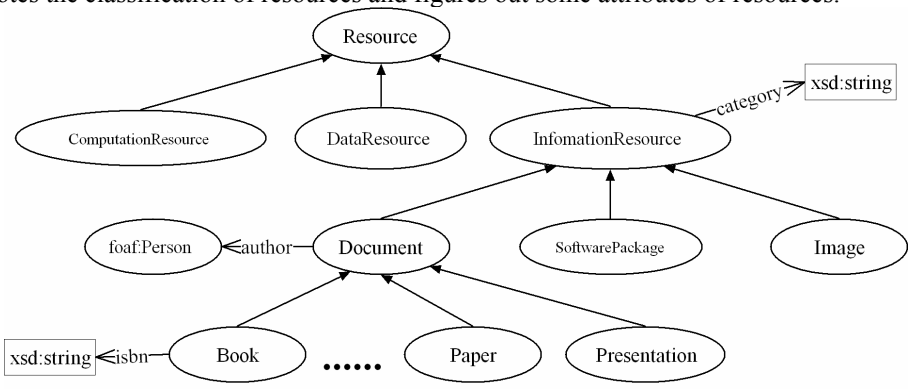

Fig. 3. Resource ontology

Currently, the ontology discussed above is relatively rough, and some entities defined in it are domain dependent. One of our future researches is to define a more general and domain independent ontology.

3.2 The rules

Besides the ontology, we give more explicit meaning to the properties by defining some rules, such as:

$$member(?z, ?x) \wedge member(?z, ?y) \wedge Person(?x) \wedge Person(?y) \rightarrow sameGroupOf(?x, ?y).$$

It means that if ?x and ?y are persons and group ?z has the member ?x and ?y, then ?x and ?y have the relationship of **sameGroupOf**. In fact, this rule gives a more clear meaning to **sameGroupOf** property. And a group may declare all its members, it can be deduced that any two of these members has the relationship of **sameGroupOf** then.

Another rule different from the one above looks like this:

$$phdSchoolmateOf(?x, ?y) \wedge phdStudentOf(?y, ?z) \rightarrow phdStudentOf(?x, ?z).$$

That is to say, if ?y is a PhD student of ?z, and ?x and ?y have the relationship of **phdSchoolmateOf** (note: the meaning of this property in the context is that they are both PhD students of one superviser), we can deduce that ?x is also a PhD student of ?z. It

means that although there do not have the explicit declaration to denote ?x is a PhD student of ?z, it can be reasoned using the exiting information and the rules.

By adding such rules, the expressive ability and reasoning ability are both enhanced commendably.

4 Case Study

Now, the application domain of policy language is WonderSpace, a Semantic Web application developed by our Lab. The basic scenario in WonderSpace is that user can upload the resources s/he wants to share with others to the server and specify access control policy about the resources using entities defined in our ontology. Whenever a user requests to access to one resource, the server will decide to accept or reject the request by using the policy and FOAF information of the users stored in server.

Here, we propose some use cases to denote how to express access control policies.

Example 1. Jack published a paper and asserted that the members of WonderSpace group could read this paper, while the members of DrSNP group could download it.

$$\text{member}(\text{wonderspace}, ?x) \rightarrow \text{isPermittedReadWith}(?x, \text{paper1}),$$
$$\text{member}(\text{drsnp}, ?y) \rightarrow \text{isPermittedDownloadWith}(?y, \text{paper1}).$$

Example 2. Colin published a presentation and asserted the access control policy like this: only my PhD students in the same group with me will be allowed to download this presentation. This policy can be expressed as follows:

$$\text{sameGroupOf}(?x, \text{colin}) \wedge \text{phdStudentOf}(?x, \text{colin}) \rightarrow \text{isPermittedDownloadWith}(?x, \text{presentation1}).$$

Example 3. One of the system policies may be announced like this: only the students of a professor can publish a book. Though it looks a little unreasonable (in some cases it will be useful), it may be represented as follows:

$$\text{technicalTitle}(?z, \text{“professor”}) \wedge \text{studentOf}(?x, ?z) \wedge \text{Book}(?y) \rightarrow \text{isPermittedPublishWith}(?x, ?y).$$

As **phdStudentOf** is the subproperty of **studentOf**, considering this rule, it can be deduced that a PhD student of a professor can also publish book.

Example 4. A student Mary uploaded a note and denoted that: my teachers and their colleagues were prohibited to read this note,

$$\text{studentOf}(\text{Mary}, ?x) \wedge \text{colleagueOf}(?x, ?z) \rightarrow \text{isProhibittedReadWith}(?x, \text{note1}) \wedge \text{isProhibittedReadWith}(?z, \text{note1}).$$

Our policy engine is under development, it is undoubtedly a hard work because of the complex reason over policies, but it is also proved an interesting work. For simplifying the problem, we suppose that both resources and related policies and the FOAF information of users are all creditable and stored in the server. When a user proposes a request to access (such as download) to one resource, the policy engine will search all related policy about the resource and verify one by one whether the user satisfy the conditions specified in the policy by using the FOAF information. Take example 2 as an

instance, if Bill requests to download presentation1, this policy will be found and the engine will look into Bill's FOAF file and try to find if there have the **sameGroupOf** and **phdStudentOf** properties related to Colin. If succeed, Bill will be allowed to download presentation1, if not (one may possibly not declare all relationship in his FOAF file), the engine will look through Colin's FOAF file and try to deduce the relationships between Bill and Colin. At the same time, because there has a rule about **sameGroupOf** defined in advance, the engine will also investigate whether the group Colin belongs has announced that both Bill and Colin are all its members. These are only basic ideas about the policy engine, it still have a lot of work for us to do.

5 Conclusion and Future Work

We have introduced an approach to design access control policies based on OWL and SWRL. Our work includes the definition of OWL ontology and some policy rules, it help users to specify who are permitted/prohibited to take what action to what resources. The most obvious feature of our work different from early work is that we use rules to help users express their access control policies. We give more formal meaning to the property defined in OWL ontology and help the engine deduce more information by adding some rules. And we also allow the users to assert their access control policies in the form of rule, which is more nature to the users, and they will express the policy more accurately because of the powerful expressive ability of rules.

As discussed earlier, the policy ontology and rules is still domain dependent in some ways. One of our future researches is to develop a more general ontology. The reasoning engine is also an important work under consideration. And we are planning to develop a full access control policy language and apply it to the no central controlled environment: it means that the information about users and resources are not controlled centrally in server, and are incomplete sometimes. It will give more obvious prominence to the language's advantage of interoperability at the level of semantics and powerful reasoning ability. Of course, it will bring other problems too, such as the resolution of policy confliction, the problem of trust. Our future work will also include how to resolve these problems.

Acknowledgments

This work is supported in part by National Key Basic Research and Development Program of China under grant 2003CB317004 and in part by JSNSF under grant BK2003001. We would like to thank the members of WonderSpace group for their suggestions on this paper.

References

1. I. Horrocks, P. F. Patel-Schneider: A Proposal for an OWL Rules Language. WWW (2004) ACM
2. I. Horrocks, P. F. Patel-Schneider, and F. Harmelen: From SHIQ and RDF to OWL: The Making of a Web Ontology Language. Journal of Web Semantics (2003)
3. I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean: SWRL: A semantic web rule language combining owl and ruleml. W3C Member Submission, 21 May 2004. Available at http://www.w3.org/Submission/SWRL/.
4. I. Horrocks, B. Parsia, P. F. Patel-Schneider, and J. Hendler: Semantic Web Architecture: Stack or Two Towers? PPSWR (2005): Accepted Papers
5. J. M. Bradshaw, S. Dutfield, P. Benoit, and J. D. Woolley:KAoS: Toward An Industrial-Strength Open Agent Architecture. Software Agents, J.M. Bradshaw (ed.), AAAI Press (1997) 375-418
6. J.B.D Joshi.: Access-control language for multidomain environments. Internet Computing, IEEE Volume 8, Issue 6, Nov.-Dec(2004) 40 - 50
7. L. Kagal, T. Finin, and A. Joshi: A policy language for a pervasive computing environment. IEEE 4th International Workshop on Policies for Distributed Systems and Networks (2003). http://citeseer.ist.psu.edu/kagal03policy.html
8. L. Kagal: Rei Ontology Specifications, Ver 2.0. http://www.cs.umbc.edu/~lkagal1/rei/.
9. N. Damianou, N. Dulay, E. Lupu, and M. Sloman: The ponder policy specification language. The Policy Workshop (2001) , Bristol U.K. Springer-Verlag, LNCS 1995
10. P.F. Patel-Schneider, P. Hayes, I. Horrocks (eds.): OWL: Web Ontology Language Semantics and Abstract Syntax. W3C Recommendation 10 February 2004. Latest version is available at http://www.w3.org/TR/owl-semantics/
11. P. Hayes (ed.):RDF Semantics. W3C Recommendation 10 February 2004. Latest version is available at http://www.w3.org/TR/rdf-mt/
12. T. Moses, Entrust Inc: eXtensible Access Control Markup Language (XACML), Ver 2.0. http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf