Annotation-Based Access Control for e-Professionals

Peyman Nasirifard and Vassilios Peristeras

Digital Enterprise Research Institute, National University of Ireland, Galway,
IDA Business Park, Lower Dangan, Galway, Ireland, firstname.lastname@deri.org

Abstract

Collaborative Working Environments (CWE) provide shared workspaces that enable eProfessionals to work together and share resources that they own. Most current shared workspaces provide coarse-grained role-based access control policies which bring the functionalities of shared workspaces under question. In this paper, we present Annotation-Based Access Control, an approach towards access control which benefits from user annotations to annotate eProfessionals using various fixed and desired open vocabularies (tags) and helps to build a more flexible access control mechanism based on relationships among eProfessionals (or people). We also present our prototype, a gadget, which we have developed to enable this access control mechanism and evaluate it.

Keywords

Access Control, Shared Workspace, Annotation, Social Network

1 Introduction

eProfessionals[1] are professionals that their work rely on Internet and telecommunication technologies. Usually eProfessionals collaborate together towards achieving a goal; e.g. completing a project. Collaborative Working Environments (CWE) provide the necessary infrastructures like shared workspaces for eProfessionals to work together in different time zones and share various resources. We have analysed the access control mechanism in some shared workspaces, such as BSCW and Microsoft SharePoint. The current mechanisms within these shared workspaces suffer from fine-granularity. In other words, users are not able to express flexible access control policies within shared workspaces. In most cases, from the system perspective, a user falls into a specific *group* (e.g. root, normal user) and s/he acquires access to all resources that are shared to that group; i.e. role-based access control. From the user perspective, a user can have several contacts and perhaps assign them some fixed roles as well.

Sharing is not limited to shared workspaces. One of the key architectural concepts of Web 2.0 is *sharing*. Sharing makes the traditional Web open to all agents to contribute. A common example is social bookmarking systems, where people save bookmarks and share them with their contacts. We have studied some social bookmarking systems, such as del.icio.us. Again the current access control mechanisms within social platforms suffer from fine-granularity as well. In other words, users can make their resources publicly available to all of their contacts (or all users) or keep them private. The possibility of sharing selectively some resources with some contacts based on specific criteria is not possible in most platforms, whereas the current technologies provide the sufficient infrastructure for that.

In this paper, we present an approach for access control by annotating people and defining access control policies based on annotations. Annotation is nowadays used in many social systems and can be used for organizing items / resources. The social systems use annotations for annotating resources, but not people. We use annotations for both people and resources. Our annotation

[1] http://en.wikipedia.org/w/index.php?title=E-professional&oldid=181983889

mechanism is based on two sets of vocabularies: One of them is a closed fixed set of vocabularies and the other is user-defined open set of vocabularies. We benefit from Semantic Web [Berners-Lee, Hendler, Lassila, 2001] technologies for annotations, storing and retrieving data. It helps us to interact with various platforms or even other applications can integrate with our platform. We also present a prototype that we have developed to test and evaluate our access control mechanism.

2 Annotation-Based Access Control Model

Access Control is the ability to deny or permit the use of some resources by some entities[2]. There exist plenty of approaches and mechanisms towards controlling the access: access control lists, role-based access control [Ferraiolo, Kuhn, 1992], [Sandhu, Coyne, Feinstein, Youman, 1996], attribute-based access control [Kolter, Schillinger, Pernul, 2007], etc. Each approach has its own advantages and disadvantages. In a shared workspace or social platform, where the people collaborate together and share resources, there should definitely exist some kind of access control mechanisms.

Annotation is a common mechanism which is used nowadays in many social softwares and also managing personal information. It is supposed that it eases the finding of desired resources. Our access control model is based on annotations. It benefits *partially* from social acquaintances to express the annotation mechanism, however it is not only limited to it and open vocabularies can be also utilized for annotations. In this approach, end users are able to annotate their contacts and define policies based on their annotations. In this case, only those annotated contacts that fulfill the required policies will have access to specified resources. A simple example follows: User A annotates user B which is part of his contacts as *supervisor*. User A owns also several resources and defines different policies for them. In this case, all resources that have *supervisor* in their policies and their policies express that they can be shared with the people with the role *supervisor*, are automatically accessible to the user B which has the role *supervisor*. These resources may vary from URLs / URIs to quick short messages.

Annotation-Based access control is very close to what we do in our real lives to share the resources we own. We may share the key of our apartments to our parents, but not to our friends. Based on this simple scenario, in Annotation-Based access control, both our parents and friends are our contacts, but our parents have been tagged as *parent* and our friends have been tagged as *friend* and so we distinguish between different contacts / connections.

Our current access control model composes of three main entities and two main concepts: *Person*, *Resource*, and *Policy* are three entities; *Annotation* and *Distance* are two main concepts. A Person is an entity with the RDF [W3C Semantic Web Activity 2004] type Person (http://uncle-share.com/ontology/Person). A Person is connected to zero or more other Persons. Each connection between Persons can be annotated with zero or more Annotations. An Annotation is a vocabulary or a set of vocabularies that are connected together and aims to describe the Person. A Person owns zero or more Resources. A Resource is an entity with the RDF type Resource (http://uncle-share.com/ontology/Resource) and is owned by (isOwnedBy) one or more Persons. Resources are in the form of URIs / URLs / short messages (A set of words that are connected together). A Resource can be either private or public. A private Resource has zero or more Policies, whereas a public resource has one or more Policies. A Policy is an entity with the RDF type Policy (http://uncle-share.com/ontology/Policy). A Policy is defined by (isDefinedBy) one Person and belongs to (belongsTo) one Resource. A Policy has one Annotation and one Distance. Again an Annotation is a vocabulary or a set of vocabularies that are connected together and aims to describe the Person that the Resource should be shared to. A Distance is a numerical value which determines the *depth* that the Policy is valid. *Depth* is

[2] http://en.wikipedia.org/w/index.php?title=Access_control&oldid=199054443

actually the shortest distance among two Persons with consideration of Annotations. Depth will be more clear with an example in following sections. A Person defines zero or more Policies. Figure 1 demonstrates the main elements of access control model.

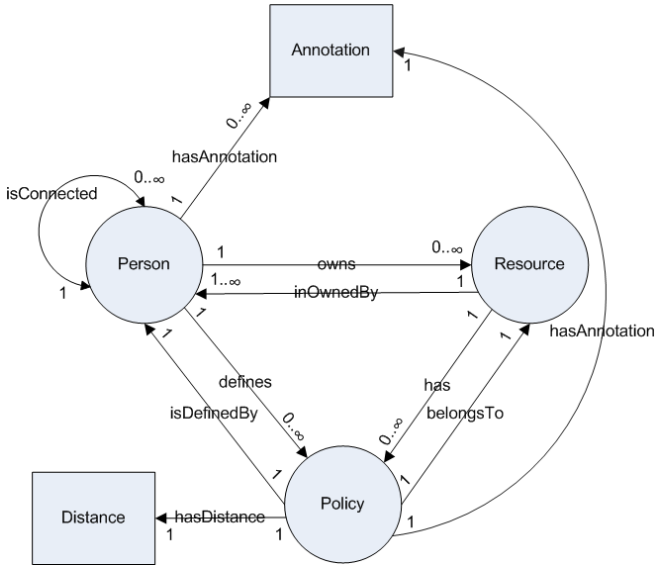

Figure 1: Main elements in access control mechanism and their relationships

3 Uncle-Share: Annotation-Based Access Control Prototype

To evaluate this approach, we have developed a prototype called *Uncle-Share* which enables Annotation-Based access control. The term *Uncle* reflects the social aspects of the system, as the users annotate their contacts with social-related vocabularies and the term *Share* reflects the access control aspects of the system. From the other hand, the combination of Uncle and Share (Uncle-Share) returns a meaningful name. In the following sections, we describe the most important parts of Uncle-Share.

3.1 Service-Oriented Architecture

In Service-Oriented Architecture [Papazoglou, Traverso, Dustdar, Leymann, 2007] (SOA), business processes are packaged as services and are accessible via end points to end users. Different applications can be built on top of this architectural paradigm. Uncle-share is based on SOA. It provides several SOAP-based services to end users. In other words, all functionalities of Uncle-Share (registration, changing password, adding persons and resources, fetching shared resources, etc.) are wrapped as Web services. Following this approach enables developers to utilize all functionalities of Uncle-Share within their own applications. Uncle-Share currently provides the following services:

- *Handle Object*: This service enables end users to register themselves to the system and/or change their passwords.
- *Handle Connection*: This service enables end users to add connections between persons; persons and resources; and persons and policies. This service enables also end users to annotate those connections with close and open vocabularies.
- *Get Connection*: This service enables end users to get who/what stuff is connected to a specific person.
- *Get Available Resources*: This service returns the available resources to a specific person based on *Distance* input. The parameter *Distance* is actually the depth that the service should dig in, in order to find the available resources.

All services accept an XML file as input which contains some required parameters. The authentication is based on the user name and password; and is done via the request within XML. The links to WSDL files, the Document Type Definition (DTD) of requests, sample requests and sample clients for accessing services are accessible online[3].

The services store and retrieve data in RDF [W3C Semantic Web Activity 2004] format. Exporting the data as RDF can be utilized in other applications or perhaps the cross-integration among applications.

3.2 User Interface: Uncle-Share Gadget

Wikipedia[4] defines widget (or control) as an interface element that a computer user interacts with, such as a window or a text box. Several widgets together build a gadget, like a messenger or calendar. These two terms, widget and gadget, are used sometimes to refer to the same object. Gadgets enable end users to have multiple applications in one page. There exist currently many gadget / widget platforms, open source and commercial gadget-building tools. NetVibes and iGoogle are two mostly used gadget platforms. Both platforms provide basic tools for building gadgets / widgets. Netvibes provides a Universal Widget API (UWA) as a way for widgets to be available on every widget / gadget platform or blog system. However, the gadgets which are built for iGoogle can be also embedded within every gadget platform or blog system.

Having the gadget as user interface enables end users to have Uncle-Share besides other applications and this can attract more users, as they should not launch a new application or browse a new Web page to utilize Uncle-Share. In particular, we decided to use iGoogle for developing our gadget, as Google provides sufficient documentations and supports; however as we mentioned, our gadget can be embedded into any widget / gadget platform or Web site. We used AJAX [Zakas, McPeak, Fawcett, 2007] (Asynchronous JavaScript and XML) for developing the user interface.

The only client-side requirement is that the browser should support JavaScript, as the gadget was developed as JavaScript. Figure 2 demonstrates the main user interface of Uncle-Share and its preliminary logo.

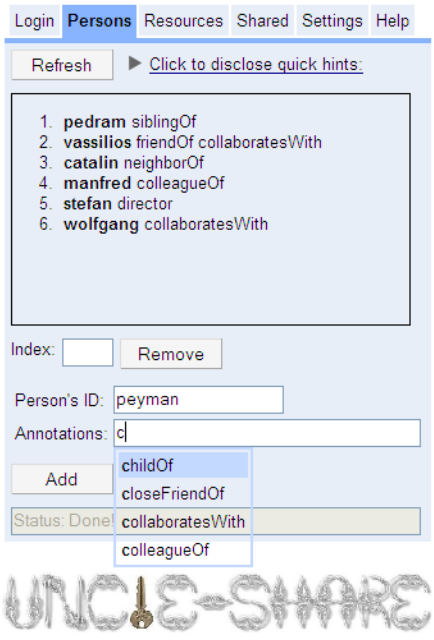

Figure 2: The Uncle-Share gadget and logo

[3] http://vmecos01.deri.ie:8081/uncle-share/index.htm

[4] http://en.wikipedia.org/w/index.php?title=GUI_widget&oldid=197085703

The gadget has six main tabs: Login, Person, Resources, Shared, Settings, and Help. All users should subscribe first via Login tab. The subscription is pretty straightforward. To keep the user interface as small as possible, due to the midget nature of gadgets, we had to cut off many fields from the registration panel. Current registration requires just full name, user name and password. Under the persons tab, users are able to add contacts, annotate them, and remove some of their contacts. Under the resources tab, users are able to add various resources (URLs / URIs / short messages) and assign different sharing policies to them. Under the Shared tab, users are able to see the resources that have been shared by others. They can set the distance to increase or decrease the scope of shared resources. Under the Settings tab, end users are able to change the server and change their passwords. Uncle-Share server (SOA server) is a Java WAR file which can be installed on any machine and end users can have their own instances of Uncle-Share SOA server. Under the help tab, there exists a link to the tutorial video and some technical and contact information regarding the platform.

For annotations and also policies, Uncle-Share has a *suggest box*. In suggest box, end users will get some recommendations / suggestions from Uncle-Share. These are considered to be fixed vocabularies and are based on RELATIONSHIP [Davis, Vitiello Jr, 2005] ontology. It is an extended version of FOAF and a set of vocabularies for describing relationships between people. Uncle-Share gadget is online and can be tried under iGoolge[5] or standalone[6].

3.3 Implementation Issues

We have chosen some specific open source and free software to implement Uncle-Share. As we mentioned, Uncle-Share stores and retrieves data in RDF. We use Sesame 2.0 as RDF store. The SOA backbone is based on Apache CXF which eases the development of Web service. Using its nice annotation mechanism within Java code, it makes it easier to develop Web service. Moreover, it generates automatically the WSDL files of Web service.

For building AJAX-based gadget, we used Google Web Toolkit (GWT). The GWT, which is a free Java package, gives us the basic useful elements of the UI, such as text boxes, buttons, tabs etc. GWT has a Java to JavaScript compiler which compiles the Java source and generates desired user interface. For more information on GWT, refer to [Hanson, Tacy, 2007].

4 Sample Scenario

In this part, we present a meaningful scenario to show how annotation-based access control and uncle-share operate. In our scenario, we have four users: Peyman, Vassilios, Stefan and Wolfgang. Peyman adds Vassilios to his contacts and annotates him with *collaboratesWith* and *friendOf*. Peyman adds also Stefan to his contacts and annotates him as *director*. Peyman owns three resources: *www.resource1.com*, *www.resource2.com* and *I_need_to_talk_to_you_please*. The latter resource looks like a short message which is also considered as a resource. Peyman defines following policies: *collaboratesWith:1* and *friendOf:1* for *www.resource1.com*; *collaboratesWith:2* and *friendOf:2* for *www.resource2.com*; and *director:1* for *I_need_to_talk_to_you_please* resource. The numerical value which comes in policies after the annotation (and a colon as the separator) is the distance that the policy will be valid. Note that Uncle-Share calculates the shortest path among two persons and checks whether it fulfills the distance requirement or not.

Vassilios adds Wolfgang to his contacts and annotates him as *collaboratesWith* and *friendOf*. He also adds Peyman and annotates him as *student*. Vassilios owns also two resources: *www.resource4.com* and *www.resource5.com*. He defines following policies for his resources:

---

[5] http://www.google.com/ig/adde?moduleurl=http://epeyman.googlepages.com/uncle-share-gadget.xml

[6] http://vmecos01.deri.ie:8081/uncleshareUI/

collaboratesWith:1 and *friendOf:1* for *www.resource4.com* and *student:1* for *www.resource5.com*. Both Wolfgang and Stefan do not add any contacts or resources. Figure 3 demonstrates the graph of the mentioned scenario. Due to the limitation of space, we have omitted the bi-directional links; mainly say from resources to persons, and from policies to resources and persons.

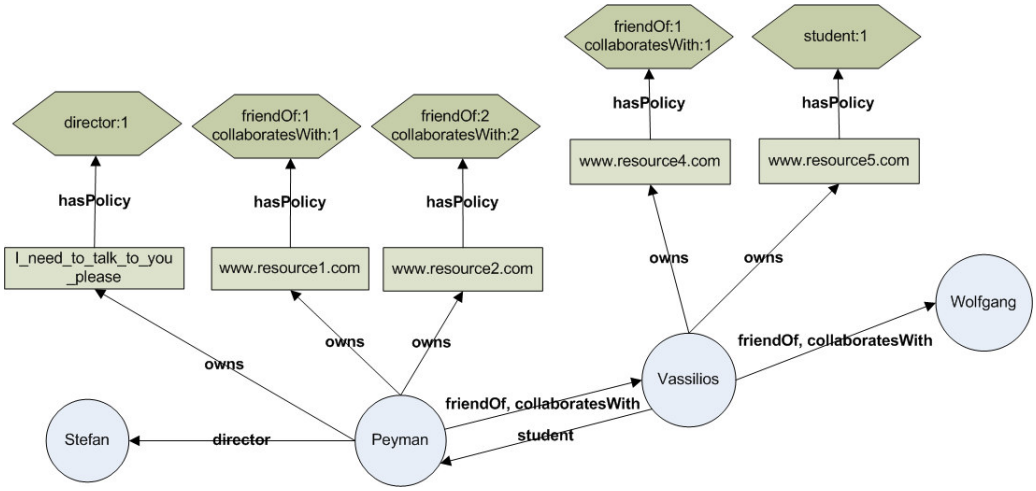

Figure 3: Sample scenario

Based on above scenario, here we will see which resources are accessible to whom. Peyman has access to his three resources and *www.resource5.com* via Vassilios, because *www.resource5.com* is accessible to the Vassilios's contacts that have been annotated as *student* and have maximum distance one to Vassilios and Peyman fulfills this policy. Vassilios has access to his two resources and also two of Peyman's resources: *www.resource1.com* and *www.resource2.com*, because he fulfills the policies. Wolfgang will see *www.resource4.com* which was shared via Vassilios to him and also *www.resource2.com* which was shared via Peyman to him. Stefan will see the short message from Peyman: *I_need_to_talk_to_you_please*.

5 Discussions and Related Works

In this section, we have an overview of related works and compare different approaches with Uncle-Share's approach. [Ryszard Kruk, Grzonkowski, Gzella, Woroniecki, Choi, 2006] suggest a role-based policy-based access control for social networks, where the access rights will be determined based on social links and trust levels between people. [Carminati, Ferrari, Perego, 2006] present the same approach. Our approach is different from their approaches in several ways. First, those approaches use fixed vocabularies and in our approach, fixed vocabularies are suggested to end users, but we do not force users to use them. End user are allowed to use open vocabularies as well as fixed vocabularies for annotations. The open vocabularies enable end users to express the trust level as well. As an example, instead of using (*friend 80%*), end users can express it with the notion of *closeFriendOf*; (*friend 50%*) can be expressed as *commonFriend* or *normalFriend* and so on. In this case, the model will be more realistic, as we don't label our friends with numerical values in real lives. We also calculate the distance between two persons with the consideration of the annotations. For example, if person A is connected to person B and this connection has the annotation *student*, in this case the distance from person A to B (directional) with the consideration of *student* is one. The distance from person A to B (directional) with the consideration of any other annotations (e.g. friendOf) is infinity. The distance from person B to A (directional) is also infinity, because person B has no outgoing link to person A.

[Carminati, Ferrari, Perego, 2007] present the idea of private relationships and the fact that due to privacy reasons, not all relationships should be public. In our model, all relationships are private,

as there is no need to publicly announce relationships between people. End users can freely publish their own relationships, if needs be.

6 Conclusion and Future Works

In this paper, we have presented the Annotation-Based access control and its prototype, Uncle-Share. Uncle-Share enables end users to annotate their contacts and set different policies for their resources based on their annotations. Our model can be seen as an extension of Role-Based access control, where people are able to define their own roles and assign them to others in a user-centric model. We are currently working to extend RELATIONSHIP and add more collaboration-based vocabularies to it. The current release of Uncle-Share is not context-aware. We plan to embed Context information into the process. We plan to build a simple mashup to fetch context information of eProfessionals from their Micro-blogs like Twitter. This can be done via defining a fixed set of vocabularies for context model or via doing simple natural language processing. One of the interesting extensions is using Open Social API to make the Uncle-Share embedded into social networks like MySpace and Orkut. Open Social follows the idea of *Write once, run anywhere* and enables developers to develop cross-platform applications among social Web sites. We plan also to check the feasibility of integration of Uncle-Share within social bookmarking systems like del.icio.us. More advanced user model and suggestions / recommendations, exporting social networks of users, and prioritizing the policies are different possible improvements. Due to the small nature of widgets / gadgets, we may develop a full-screen version of user interface and put a snippet of the main interface into the gadget.

Acknowledgement

This work has been partly funded by Ecospace (Integrated Project on eProfessional Collaboration Space) project: FP6-IST-5-35208 and Lion project supported by Science Foundation Ireland under Grant No. SFI/02/CE1/I131

References

Berners-Lee, T.; Hendler, J.; Lassila, O.: The Semantic Web, A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. Scientific American, May 2001.

Carminati, B.; Ferrari, E.; Perego, A.: Rule-Based Access Control for Social Networks. In OTM Workshops (2), pages 1734–1744. Lecture Notes in Computer Science, Springer-Verlag, Berlin Heidelberg New York, 2006.

Carminati, B.; Ferrari, E.; Perego, A.: Private Relationships in Social Networks. In Proceedings of ICDE Workshops, pages 163–171, 2007.

Davis, I.; Vitiello Jr, E.: RELATIONSHIP: A vocabulary for describing relationships between people. WWW page. http://vocab.org/relationship/, 2005, accessed 20-March-2008.

Ferraiolo, D.F.; Kuhn, D.R.: Role Based Access Control. In 15th National Computer Security Conference, 1992.

Hanson, R.; Tacy, A.: GWT in Action: Easy Ajax with the Google Web Toolkit. Manning Publications Co., Greenwich, CT, USA, 2007.

Kolter, J.; Schillinger, R.; Pernul, G.: A Privacy-Enhanced Attribute-Based Access Control System. In DBSec, volume 4602 of Lecture Notes in Computer Science, pages 129–143. Springer, 2007.

Papazoglou, M.P.; Traverso, P.; Dustdar, S.; Leymann, F.: Service-Oriented Computing: State of the Art and Research Challenges. In Computer, volume 40, pages 38–45, 2007.

Ryszard Kruk, S.; Grzonkowski, S.; Gzella, A.; Woroniecki, T.; Choi, H.C.: D-FOAF: Distributed Identity Management with Access Rights Delegation. In Proceedings of Asian Semantic Web Conference (ASWC), pages 140–154, 2006.

Sandhu, R.S.; Coyne, E.J.; Feinstein, H.L.; Youman, C.E.: Role-Based Access Control Models. In IEEE Computer 29(2), 1996.

W3C Semantic Web Activity: Resource Description Framework (RDF). WWW page. http://www.w3.org/RDF/, 2004, accessed 20-March-2008.

Zakas, N.C.; McPeak, J.; Fawcett, J.: Professional Ajax (Programmer to Programmer). Wiley Publishing, second edition, 2007.

FOAF: The Friend of a Friend (FOAF) project. WWW page. http://www.foaf-project.org/, accessed 20-March-2008.

Twitter. WWW page. http://twitter.com/, accessed 20-March-2008.

Open Social: Open Social Initiative. WWW page. http://opensocial.org/, accessed 20-March-2008.

Sesame RDF Store, version 2.0. WWW page. http://www.openrdf.org/, accessed 20-March-2008.

Apache CXF. WWW page. http://incubator.apache.org/cxf/, accessed 20-March-2008.

Google Web Toolkit. WWW page. http://code.google.com/webtoolkit/, accessed 20-March-2008.

NetVibes. WWW page. http://www.netvibes.com/, accessed 20-March-2008.

iGoogle. WWW page. http://www.google.com/ig, accessed 20-March-2008.

Universal Widget API, NetVibes. WWW page. http://dev.netvibes.com/, accessed 20-March-2008.

BSCW: Basic Support for Cooperative Work (BSCW). WWW page. http://www.bscw.de/, accessed 20-March-2008.

Microsoft SharePoint. WWW page. http://www.microsoft.com/sharepoint/default.mspx, accessed 20-March-2008.

del.icio.us: Online Bookmarking System. WWW page. http://del.icio.us/, accessed 20-March-2008.

Orkut: Social networking and discussion site operated by Google. WWW page. http://www.orkut.com/, accessed 20-March-2008.

MySpace: An international site that offers email, a forum, communities, videos and weblog space. WWW page. http://www.myspace.com/, accessed 20-March-2008.