Proceedings of the First International Workshop on

Trust and Privacy on the Social and Semantic Web (SPOT 2009)

co-located with the
6th Annual European Semantic Web Conference
June 1, 2009, Heraklion, Crete, Greece

Editors

Michael Hausenblas

Philipp Kärger

Daniel Olmedilla

Alexandre Passant

Axel Polleres

Published online as CEUR Workshop Proceedings,
ISSN 1613-0073, CEUR-WS.org/Vol-447/

Preface

The goal of the Semantic Web is the publishing and sharing of semantically annotated data. As a result, machines are able to understand and process such data in order to enhance and increase the tasks that can be performed on today's Web. In the last years the Semantic Web has become more and more social, that is, such data does not anymore consist only of business services and products but it increasingly includes information about people, their relationships with others, what they do, believe and like. Semantic Web technologies are used to ease and increase communication and information exchange on the Web, and therefore interlinking various sources of distributed data. For instance, efforts like FOAF and SIOC provide the ability to merge social networks and to exchange socially-created data among different platforms and providers. At the same time, Semantic Web technologies have been advanced and currently numerous knowledge repository are exposed to the public, following the Linking Open Data movement.

Naturally, the general goal of combining distributed semantically annotated knowledge raises issues of trust and privacy. For example, information gathered from the Semantic Web shall be trusted before it is further processed. Also social activities empowered with semantic technology like social networking, blogging, desktop or resource sharing require privacy to be ensured. In many cases, information about a person is not published by herself, but by others, therefore possibly risking a user's privacy[1]. Additionally, once it is published, not only a reader of such information should decide whether it is to be trusted and if yes, to what degree, it is also a question of how such information may be used, and whether it may attempt against other people's privacy[2].

Trust and Privacy are needed both in terms of publishing and consuming data: "which sources should I trust?", "how to ensure that this information is valid?", "does it really come from someone I know or an authoritative source?", "how to share part of my identity only to trusted people?", or "how to ensure the information I disclose will not be misused or responsible will be made accountable?" are questions that need to be answered now while more and more people use the Semantic Web for socializing, for sharing personal data, and for general information exchange.

Semantic Web technologies have reached a status where they influence our daily lives. On the one hand, applications for sharing semantically annotated pictures, blogs, and videos as well as semantic-enhanced social networking platforms are present. On the other hand, the so-called Web of Data with its thousands of billions of triples is leaving its research prototype status. Applications using

[1] an example for this can be found at `http://www.smh.com.au/news/technology/virgin-sued-for-using-teens-photo/2007/09/21/1189881735928.html`

[2] for an example, see `http://www.washingtonpost.com/wp-dyn/content/article/2007/11/29/AR2007112902503.html`

Semantic Web technologies start to arise and to be used by a large number of users. However, although trust and privacy play a crucial role in its final development and adoption, in most of the running systems and research prototypes no or not sufficient solutions to address these topics are considered. The Semantic Web as well as the Social Web has reached a state where those issues have to be addressed seriously in order to become reality.

The First International Workshop for Trust and Privacy on the Social and Semantic Web (SPOT2009) brings together, among others, researchers and developers from the field of Semantic Web, the Social Web, and trust and privacy enforcement. It provides the opportunity to discuss and analyze important requirements and open research issues for a trustful Semantic Web.

We are grateful to the members of the Program Committee of SPOT2009 for their support. We would like to thank all the authors who submitted their work for their interesting contributions. Further on, we thank Prof. Piero Bonatti for being available as invited speaker and, last but not least, the European COST Action IC0801 "Agreement Technologies" for supporting the invited talk.

May 2009

Michael Hausenblas
Philipp Kärger
Daniel Olmedilla
Alexandre Passant
Axel Polleres
Organizing Committee of SPOT2009

Organization

SPOT 2009 is co-located with the European Semantic Web Confenference 2009 (ESWC).

Organizing Committee

Michael Hausenblas - DERI, Galway, Ireland
Philipp Kärger - L3S Research Center, Hannover, Germany
Daniel Olmedilla - Telefonica R&D, Madrid, Spain
Alexandre Passant - DERI, Galway, Ireland
Axel Polleres - DERI, Galway, Ireland

Program Committee

Vinicius Almendra - ADDLabs, Brasil
Chris Bizer - Freie Universität Berlin, Germany
Piero Bonatti - University of Naples, Italy
John Breslin - DERI, NUI Galway, Ireland
Dan Brickley - FOAF Project, UK
Juri L. De Coi - L3S Research Center, Germany
Stefan Decker - DERI, NUI Galway, Ireland
Fabien Gandon - INRIA Sophia-Antipolis, France
Wolfgang Halb - Joanneum Research, Austria
Harry Halpin - University of Edinburgh, UK
Michael Hausenblas - DERI, NUI Galway, Ireland
Tom Heath - Talis, UK
James Hendler - Rensselaer Polytechnic Institute, USA
Bettina Hoser - Universität Karlsruhe, Germany
Lalana Kagal - MIT CSAIL, USA
Philipp Kärger - L3S Research Center, Germany
Daniel Olmedilla - Telefonica R&D, Spain
Sascha Ossowski - University Rey Juan Carlos, Spain
Alexandre Passant - DERI, NUI Galway, Ireland
Axel Polleres - DERI, NUI Galway, Ireland
Simon Schenk - University of Koblenz, Germany
Carles Sierra - IIIA-CSIC, Spain
Henry Story - Sun Microsystems, France
Alessandra Toninelli - DEIS, Università di Bologna, Italy

Additional Referees

Ramon Hermoso
Sergej Zerr

Sponsoring Organizations

Cost Action IC0801 on Agreement Technologies

Keynote

Prof. Piero Bonatti

Trust and Privacy on the Social and Semantic Web
(SPOT 2009)

A Comparison of Terminological and Rule-based Policy Languages

Piero A. Bonatti

Università di Napoli Federico II

Security and privacy policies commonly consist of declarative constraints over resource usage (data and services). Therefore logic-based representation languages are well-suited as a foundation of policy languages. Indeed, the semantics of standard languages like XACML can be reformulated in a logic-based fashion similar to the encoding adopted in [2]; moreover, both description logics and logic programming languages (i.e., the two main families of knowledge representation formalisms) have been proposed as policy languages, see KAOS, [7], REI [5], RT [6], Cassandra [1], PeerTrust [4], and PROTUNE [3] just to name a few approaches.

Policy-related processing involves several different reasoning tasks over the axioms that constitute a policy:

- An authorization A is granted iff A is *entailed* by the policy;
- In trust negotiation, a set of credentials C unlocks a resource R iff C and the policy together entail the authorization to use R; the process of finding the sets C that enjoy this property (given the desired authorization for R) is called *abduction*;
- Usability, awareness, and validation issues make it very important to support *explanation facilities* such as those supplied by expert systems; explanation facilities convert axioms and proofs into natural language text understandable by people with no specific training in knowledge representation or computer science; when such documentation is produced automatically, it is guaranteed to be always aligned with the policy actually applied by the system; moreover, automated explanation facilities can produce *contextualized* documentation, relative to specific transactions;
- A natural privacy-related operation is *comparing* the privacy policy published by a web site with the privacy preferences of a user; the relevant question here is whether the information disclosures permitted by the web site's policy will always be permitted also by the user's privacy policy. Policy comparison can also be useful in assessing the results of a policy update; it can answer the question of whether the new policy is more permissive or more restricted than the old one.

In this talk we will assess different knowledge representation formalisms as policy languages for security and privacy, taking into account not only the kind of

constraints that they can express on resource usage, but also the degree to which the above reasoning tasks can be supported. We will conclude that currently rule-based languages are more mature than description logics as far as the general needs of security and privacy policy languages are concerned.

References

1. Moritz Y. Becker and Peter Sewell. Cassandra: Distributed access control policies with tunable expressiveness. In *5th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2004)*, pages 159–168, Yorktown Heights, NY, USA, June 2004. IEEE Computer Society.
2. Piero Bonatti, Sabrina De Capitani di Vimercati, and Pierangela Samarati. An algebra for composing access control policies. *ACM Trans. Inf. Syst. Secur.*, 5(1):1–35, 2002.
3. Piero A. Bonatti and Daniel Olmedilla. Driving and monitoring provisional trust negotiation with metapolicies. In *6th IEEE Policies for Distributed Systems and Networks (POLICY 2005)*, pages 14–23, Stockholm, Sweden, June 2005. IEEE Computer Society.
4. Rita Gavriloaie, Wolfgang Nejdl, Daniel Olmedilla, Kent E. Seamons, and Marianne Winslett. No registration needed: How to use declarative policies and negotiation to access sensitive resources on the semantic web. In *1st European Semantic Web Symposium (ESWS 2004)*, volume 3053 of *Lecture Notes in Computer Science*, pages 342–356, Heraklion, Crete, Greece, May 2004. Springer.
5. Lalana Kagal, Timothy W. Finin, and Anupam Joshi. A policy language for a pervasive computing environment. In *Policies for Distributed Systems and Networks, 2003. Proceedings. POLICY 2003. IEEE 4th International Workshop on*, pages 63–74. IEEE Computer Society, June 2003.
6. Ninghui Li, John C. Mitchell, and William H. Winsborough. Design of a role-based trust-management framework. In *IEEE Symposium on Security and Privacy*, pages 114–130, 2002.
7. A. Uszok, J. Bradshaw, R. Jeffers, N. Suri, P. Hayes, M. Breedy, L. Bunch, M. Johnson, S. Kulkarni, and J. Lott. Kaos policy and domain services: Toward a description-logic approach to policy representation, deconfliction, and enforcement. In *Policies for Distributed Systems and Networks, 2003. Proceedings. POLICY 2003. IEEE 4th International Workshop on*, pages 93–96. ACM Press, June 2003.

Trust Models for the Social and Semantic Web

Trust and Privacy on the Social and Semantic Web (SPOT 2009)

QUATRO Plus: Quality You Can Trust?

Phil Archer[1], Elena Ferrari[2], Vangelis Karkaletsis[3], Stasinos Konstantopoulos[3], Antonis Koukourikos[3], and Andrea Perego[2]

[1] i-sieve technologies, Athens, Greece
phil@i-sieve.com
[2] DICOM, Università degli Studi dell'Insubria, Varese, Italy
{elena.ferrari,andrea.perego}@uninsubria.it
[3] IIT, NCSR 'Demokritos', Athens, Greece
{vangelis,konstant,kukurik}@iit.demokritos.gr

Abstract. The QUATRO Plus project, a follow on from the original QUATRO Project, aims to balance the wisdom of the crowds with the knowledge of the experts. It uses a mixture of authenticated data sources and the opinions of end users expressed through social networking software to build a dataset that is authoritative and trustworthy. The dataset describes online resources using RDF with the upcoming W3C Recommendation, POWDER, as the underlying transport and storage mechanism. Data can be added to or queried through a variety of tools provided by the project, some of which are described in detail in this paper.

1 Introduction

There is a great deal of opinion online that is an expression of 'the wisdom of the crowds,' the classic example of this being Wikipedia. Much of this material is recognised as being extremely good. Mistakes and occasional deliberate falsehoods do occur however—something that Wikipedia itself recognises.[4] On the other hand, there are many experts whose opinions are expressed on the Web and whose opinions are open to authentication: trustmark operators. Well known examples of this include TRUSTe, Health on the Net (HON), and VeriSign.

The Quality Content And Description project (QUATRO Plus)[5] seeks to balance the wisdom of the crowds with the wisdom of the experts. QUATRO Plus follows on directly from an earlier project and has been instrumental in the development of the new W3C Protocol for Web Description Resources (POWDER). This paper presents the QUATRO Plus project with particular emphasis on its creation of POWDER documents, exposure of their provenance and their potential trustworthiness based on the reputation of the individual or organisation that created them.

The rest of this paper is organised as follows. Section 2 briefly introduces the POWDER specification and especially its trust and authentication features. Section 3 outlines the architecture of the QUATRO Plus platform and discusses

[4] See http://en.wikipedia.org/wiki/Wisdom_of_the_crowds.
[5] Project website: http://www.quatro-project.org/

```
1  <powder xmlns="http://www.w3.org/2007/05/powder#"
       xmlns:ex="http://example.org/vocab#">
2    <attribution>
3      <issuedby src="http://authority.example.org/company.rdf#me" />
4      <issued>2007-12-14T00:00:00</issued>
5    </attribution>
6    <dr>
7      <iriset>
8        <includehosts>example.com</includehosts>
9      </iriset>
10     <descriptorset>
11       <displaytext>Everything on example.com is red and square</displaytext>
12       <displayicon src="http://authority.example.org/icons/red-square.png" />
13       <ex:colour rdf:resource="http://rgb.org/vocab#red" />
14       <ex:shape>square</ex:shape>
15     </descriptorset>
16   </dr>
17 </powder>
```

Fig. 1. A simple POWDER document

how it serves metadata, whereas Sect. 4 focuses on one of its main metadata creation components, the Quality Social Network. Finally, Sect. 5 discusses related work, whereas Sect. 6 concludes the paper and outlines future research directions.

2 The POWDER Specification

The *Protocol for Web Description Resources* (POWDER) is a general purpose content and quality labelling protocol based on Semantic Web technologies, developed by the W3C POWDER Working Group.[6]

POWDER was designed as a practical means of adding RDF to collections of resources. To take a simple example, POWDER expresses statements such as 'everything on example.com is red and square; this statement was asserted by the entity described at http://authority.example.org/company.rdf#me on 14th December 2007.' This statement is exemplified in Fig. 1.

At the core of POWDER is the *Description Resource* (DR) [1], an association between:

- the DR *scope*, a set of resources [2]. Scope is expressed as an iriset, a series of restrictions on the IRIs of resources that are in scope (lines 7-9 in Fig. 1).
- a *formal description* of all resources in scope (lines 13 & 14 in Fig. 1). This is expressed by the descriptorset, which comprises RDF properties defined in external vocabularies (as ex:colour and ex:shape at lines 13 & 14 in Fig. 1). And

[6] The POWDER Working Group was chartered in March 2007 and is due to complete its work in June 2009. More details on the WG and the specification are available at http://www.w3.org/2007/powder/.

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <rdf:RDF xmlns="http://www.w3.org/2007/05/powder-s#"
       xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
       xmlns:ex="http://example.org/vocab#">
3   <rdf:Description rdf:about="http://www.example.com/foo/">
4     <text>Everything on example.com is red and square</text>
5     <logo rdf:resource="http://authority.example.org/icons/red-square.png"/>
6     <ex:colour rdf:resource="http://rgb.org/vocab#red" />
7     <ex:shape>square</ex:shape>
8   </rdf:Description>
9 </rdf:RDF>
```

Fig. 2. The output of the *describe*(u, D) function, where D set includes as element the POWDER document in Fig. 1 and u = http://www.example.com/foo/. Note that properties text and logo map from elements displaytext and displayicon, respectively, in Fig. 1.

– a *textual* and/or *pictorial summary* of the formal description. This is expressed as descriptorset members using the POWDER-defined displaytext and displayicon elements (lines 11 & 12 in Fig. 1).

Furthermore, POWDER documents associate *attribution* blocks (lines 2–5 in Fig. 1) with one or more DR blocks. Attribution blocks provide information that can be used assess the trustworthiness of the DRs in a document, including the labelling authority that created it, creation time, validity period, and so on.

Note that POWDER attribution blocks have the only purpose of denoting the authorship of a given POWDER document, thus making users able to verify its authenticity, and to decide its degree of trustworthiness. Such an approach is more specific with respet to the one of the Open Provenance Model (OPM) [3], which aims at providing a detailed description of the creation, usage, and derivation of a given artifact along its whole lifecycle. However, although OPM addresses issues that are not requirements according to the POWDER specification, such technology can be effectively applied to POWDER in order to document the history of POWDER documents.

Finally, POWDER documents can be explicitly associated with the resources they apply to by using a variety of methods. For this purpose, the describedby relationship has been defined to be used in (X)HTML link elements, HTTP `Link` headers [4], ATOM entries [5], and RDFa [6]. As illustrated in Sect. 2.1, POWDER processors allow one to apply arbitrary POWDER documents to a resource by specifying the IRI of a specific POWDER document, or by accessing a repository of POWDER documents. For a detailed description of all the possible options, we refer the reader to Sect. 4 of [1].

2.1 Processing POWDER Documents

POWDER processors implement a *describe*(u, D) function that returns an rdf:Description of resource u given a set D of POWDER documents (see Fig. 2 for an example).

```
@prefix owl:    <http://www.w3.org/2002/07/owl#> .
@prefix ex:     <http://example.org/vocab#> .
@prefix rdfs:   <http://www.w3.org/2000/01/rdf-schema#> .
@prefix wdrs:   <http://www.w3.org/2007/05/powder-s#> .
@prefix xsd:    <http://www.w3.org/2001/XMLSchema#> .

<> a owl:Ontology;
   wdrs:issued "2007-12-14T00:00:00";
   wdrs:issuedby <http://authority.example.org/company.rdf#me> .

_:iriset_1 a owl:Class; owl:equivalentClass [
   a owl:Class;
   owl:intersectionOf  (
      [ a :Restriction; owl:onProperty wdrs:matchesregex;
        owl:hasValue
          "\:\/\/(([^\/\?\#]*)\@)?([^\:\/\?\#\@]+\.)?(example\.com)(:([0-9]+))?\/"^^xsd:string; ] ) ] .
_:descriptorset_1 a owl:Class;
   rdfs:subClassOf [ a owl:Class;
      owl:intersectionOf (
         [ a owl:Restriction; owl:onProperty ex:color; owl:hasValue <http://rgb.org/vocab#red> ]
         [ a owl:Restriction; owl:onProperty ex:shape; owl:hasValue "square" ]
      ) ];
   wdrs:logo <http://authority.example.org/icons/red-square.png>;
   wdrs:text "Everything on example.com is red and square" .

_:iriset_1 rdfs:subClassOf _:descriptorset_1 .
```

Fig. 3. The OWL/RDF transform of the POWDER document in Fig. 1 in N3 syntax [9,10]. Note how attribution is expressed as owl:Ontology annotations and POWDER-defined descriptors as descriptorset annotations; POWDER-defined properties are instances of owl:AnnotationProperty.

POWDER documents can, to a large degree, be processed entirely as XML but are, in fact, transporting OWL/RDF graphs that can be accessed by performing a transformation on the XML (see Fig. 3 for an example). In *Semantic POWDER* (POWDER-S), DRs express an rdfs:subClassOf relation between the class of resources that fall within scope and the class of resources that have the given RDF description. The POWDER document as a whole is an owl:Ontology instance containing a number of such assertions, and the attribution block expresses owl:AnnotationProperty triples with the owl:Ontology instance as subject. This achieves the same goal as Named Graphs [7] as used for example by Bizer and Cyganiak in [8]: the explicit declaration of provenance information, but within existing RDF/OWL specifications.

A semantic extension bridges abstract (but named) resources and the string representations of their IRIs. This extension reflects the basic premise of POWDER, absent from RDF semantics, that the structure of a resource's IRI is a property of the corresponding resource, upon which inference can be drawn; this allows POWDER documents to assert propositions about 'all resources on example.com.'

POWDER-S offers processors the option of using standard semantic inference and querying tools to process POWDER/XML documents. The process of transforming POWDER/XML documents into POWDER-S documents is detailed in the POWDER specification [11] and also implemented as an XSLT script [12] associated with the POWDER namespace via GRDDL [13]. Together with the fact that POWDER processors respond with RDF descriptions even when only processing XML, this transformation makes POWDER tools fully compatible with other Semantic Web tools and technologies.

```
<foaf:Organization rdf:ID="me">
 <foaf:homepage rdf:resource="http://authority.example.org/" />
 <foaf:name>The Exemplary Description Authority</foaf:name>
 <foaf:nick>EDA</foaf:nick>
 <wdrs:authenticate rdf:resource="http://authority.example.org/auth.html" />
</foaf:Organization>
```

Fig. 4. A simple FOAF Agent

2.2 Attribution and Authentication

Trust is a human quality and is rarely deterministic in an absolute sense; whether trust is conferred on any data is always a balance between the likelihood of it being true and the consequence of it being false. If a DR says that a particular cake recipe is really good but following it produces an inedible lump, little harm is done. If a DR states that some medical advice is peer reviewed and can be used as the basis for diagnosis and treatment, then the consequences of falsehood are clearly much more serious and a more robust authentication method is appropriate.

Although the POWDER specification is very detailed on the form and meaning of POWDER documents and the output of conforming processors, it is deliberately non-prescriptive about trust and authentication methods; any method can be defined by a DR publisher that is appropriate for the particular context (the choice of method made by the publisher may itself be a factor in a user's assessment of trust!). What POWDER does offer is a detailed specification of how to endow DRs with attribution and authentication *credentials*, the raw information upon which a variety of trust methodologies can operate.

The issuedby attribution element is mandatory for all POWDER documents and provides the IRI that identifies a description of the entity that created the document and that is therefore responsible for the claims made in the DR(s) it contains. The strong recommendation is that such descriptions are provided as instances of the Agent class in either of the widely used FOAF [14] or Dublin Core [15] vocabularies.[7]

An RDF-aware system can use the IRI to retrieve the properties of the creator (see Fig. 4 for an example) and use them to decide if the document is trustworthy. Besides whatever properties are defined by the vocabulary used to describe the creator, POWDER also defines the authenticate property that points to a resource that gives information on how to authenticate any POWDER documents that were created by this Agent. Such a resource may be a machine readable document, such as a WSDL[8] file, that can be read and acted upon automatically. However, it is equally valid for the resource to be a simple text document that needs to be read by a person who then implements a suitable system to take advantage of the available authentication mechanism. This approach is in

[7] foaf:Agent is defined at http://xmlns.com/foaf/spec/#term_Agent. dcterms:Agent is defined at http://dublincore.org/documents/dcmi-terms/#classes-Agent.

[8] Web Services Description Language. See [16].

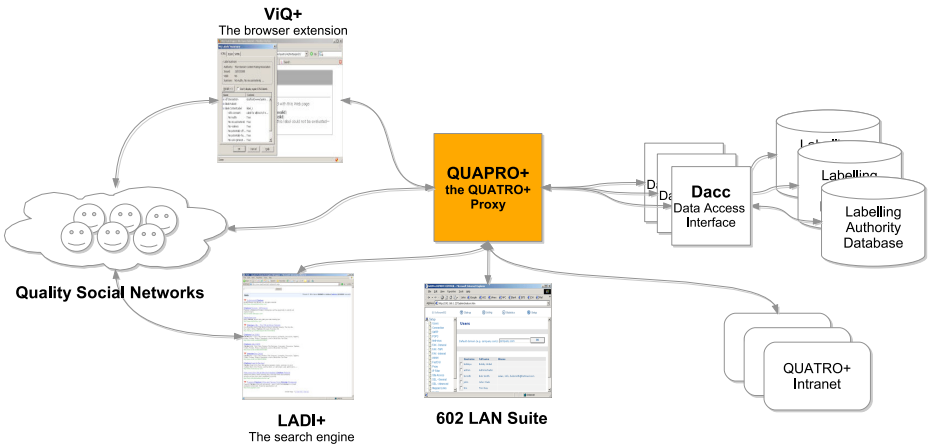

Fig. 5. The architecture of the QUATRO Plus platform

tune with the POWDER paradigm as the aim is to provide strong assurance of the identity of the creator of a given description. Such a data source will almost always require a human stage in assessing trustworthiness. Furthermore, as already noted, trust is very context-sensitive so that prescribing a single authentication mechanism would have been inappropriate. Section 3 discusses the particular authentication system used by QUATRO Plus.

There are other features of POWDER designed to facilitate trust in the data. These include certification, that is, where a DR has a scope of exactly one resource and a descriptor set that includes either or both of two descriptors: certified, which has a range of xsd:boolean, and sha1sum, the value of which is a SHA-1 checksum of the described resource. Using these, it's possible for a DR to certify the veracity of another. Another feature designed to increase the trust that can be placed in a POWDER document is the supportedby property. This can link to any form of data published by a third party that agrees with the assertions in the DR. This data may be in any form, the assumption being that an application will be built with a specific data source in mind. As an example, a POWDER document may assert that a website is suitable for children. Supporting evidence might come from a commercial content classification service.

3 Working with Trustmarks: The QUATRO Proxy

The full suite of QUATRO Plus software (whose architecture is depicted in Fig. 5) includes a search result annotation tool, LADI+ and a browser extension, ViQ+, both of which recognise the presence of links to DRs and provide authentication information to end users. A further tool, the Label Management

Environment (LME), was developed under a related project, MEDIEQ.[9] In this paper, we focus on two components of QUATRO Plus suite: The QUATRO Proxy and the Quality Social Network (QSN).

The QUATRO PROXY is a set of Web services provided by multiple servers that, among other functionalities, is able to describe resources by locating, processing and authenticating POWDER documents. Given the IRI of a Web resource, QUATRO PROXY locates relevant POWDER documents, authenticates them and uses them to compose a description of the resource. The description is returned to the client along with details of the provenance and authenticity of the data.

As mentioned earlier in this section, relevant POWDER documents are located by implementing a cascade of discovery techniques. In increasing order of efficiency these are: by checking the website itself and following links to POWDER documents (using the 'describedby' relationship type); by following a link to a POWDER document provided in the request to the proxy made by the client; and by looking the site up directly in the trustmark operators' databases to which it has access. Except in the latter case, once the relevant POWDER documents have been identified, they are authenticated by looking them up in the relevant trustmark operator's database—an automated version of the 'click to verify' model operated by all online trustmarks.

The security of the transactions and the authenticity of the server and the databases to which the proxy connects are guaranteed via message signing and SSL tunnelling. Equally importantly, trustmark operators can be confident that their data is secure.

4 Creating Descriptions: The Quality Social Network

Recent years have seen an increasing diffusion of *Web-based social networks* (WBSNs) giving their members the ability to specify and share metadata concerning online resources. Examples of such WBSNs are del.icio.us (http://del.icio.us), RawSugar (http://rawsugar.com), Flickr (http://flickr.com), and Last.fm (http://last.fm), which support the so-called *social* or *collaborative tagging*.

Although, so far, users' tags and content providers' content labels have evolved separately, we think it desirable that a convergence is established between them. The ideas underlying collaborative tagging may contribute in addressing some of the main open issues which prevented the success of content labelling. In particular, existing content labels describe a very limited subset of the Web, thus making impossible to widely exploit their advantages in term of information filtering and discovery. Moreover, content labels must be consistent with the resources they describe. However, since Web resources may change very frequently, this requires an effort which is not acceptable to content providers. In such a scenario, collaborative tagging can make any end user a possible label author, thus increasing not only the number of labelled resources, but also the

[9] MEDIEQ focused on the labelling of medical resources. Please see http://medieq.org/ for more details.

number of labels associated with the same resource. Thanks to this, it would be simpler to verify whether a label actually describes the associated resource, and it would be also possible to assess the trustworthiness of a given description by statistically analysing the whole set of labels associated with a resource.

Based on these considerations, the QUATRO Plus platform includes a social networking service, referred to as *Quality Social Network* (QSN), giving its members the ability to specify user-defined POWDER documents (referred to as *labels*) and *ratings*. As any POWDER document, besides containing a set of descriptors (the DR descriptorset component), a label states who has created it (POWDER attribution block), the set of resources it applies to (the DR iriset), and its validity period (corresponding to the validfrom and validuntil attribution elements available in POWDER). By contrast, ratings are used by WBSN members to endorse part or all the descriptors in an existing label, or to express their disagreement about the claims it contains. Three types of ratings are supported: *positive* (agreement), *negative* (disagreement), and *neutral* ('I don't know'). The trustworthiness of the descriptors contained in a label is then statistically determined based on the number of positive/negative/neutral ratings associated with them. The results of such an evaluation are made available to the other components of the QUATRO Plus architecture, and to their end users.

QUATRO Plus plans to use these features as a basis to enforce personalised access to Web resources through the support for *user preferences*. User preferences give end users the ability to state which action should be performed by their user agent upon detection of resources carrying labels with given descriptors and trustworthiness. For instance, an end user may ask to be notified when a given resource does or does not satisfy a given set of requirements concerning medical resources, or decide that the user agent must block access to a resource if more than 20% of the existing labels state that it contains pornographic content.

It is important to note that, although a publicly accessible QSN will be set up,[10] the QSN is a service which can be installed and run by any institution/organisation that wishes to set up a social network supporting collaborative labelling and rating. For this reason, the QSN gives system administrators the ability to configure the service depending on the requirements of the institution/organisation running it.

In the following section, we provide an overview of the main features supported by the QSN. The work reported here is a partial implementation of the multi-layer personalisation framework we have proposed in [17], which provides the foundations of the approach adopted in QUATRO Plus for supporting Web access personalisation through the use of Semantic Web technologies and social networking.

4.1 Labels and Ratings Evaluation and Aggregation

The main purpose of supporting collaborative labelling and rating of Web resources, is to statistically analyse the labels and ratings specified by WBSN

[10] A prototype version of the QSN, accessible only by invited members, is currently available at http://dawsec.dicom.uninsubria.it/qui/

members and identify the most objective descriptions of a resource which may then be used for content/service personalisation. In order to achieve this, the labels associated with a resource are aggregated, and for each descriptor they contain, a *trust value* is computed.

There are several different formulae which may be used for trust computation, each addressing the requirements of given application domains, granting different degrees of accuracy, and having different computational costs. For these reasons, we plan to support a set of trust computation algorithms, among which QSN administrators can choose the one which is most suitable to their purposes, also taking into account the trade-off between efficiency and accuracy.

Basically, we plan to support two main trust computation solutions. According to the basic option, all QSN members as equally trustworthy. Consequently, the trustworthiness of a label is determined only by the more or less wide consensus on the claims it contains. Such consensus dependes on the possible existence of multiple occurrences of the same label, specified by different authors, and on the positive, negative, and neutral ratings associated it. All this information is then used to compute a trust score for a given label. There exist several algorithms proposed for reputation systems [18], such as Eigentrust [19] and PeerTrust [20], which might provide a suitable solution.

Such an approach can be enhanced by associating each QSN member with a reputation score, which can then be used to assign a specific weight to his/her labels and ratings. This is an issue that has been thoroughly investigated by recommender systems [21], among which there exist examples of online communities, such as MovieLens [22] and MyWOT (http://mywot.com).

Reputation can determine the trustworthiness of a given QSN member for all the users in a community. However, it may be often the case that a given member A considers more trustworthy the opinions of member B than the ones of member C, even though the reputation of C is higher than the one of B. Recommender systems have addressed such issue by computing a similarity measure between user profiles. More recently, a different approach has been adopted by a few WBSNs. For instance, FilmTrust [23] and Moleskiing [24] propose algorithms which make use of trust relationships explicitly established between WBSN members to predict the degree of trust existing even between members not directly connected. Finally, in an earlier paper [17], we have explored also the possibility of using *trust policies*, thanks to which a given QSN member can denote the set of users he/she considers as trustworthy based on either their identity or characteristics.

4.2 Privacy Issues

Labels and ratings are not sensitive by themselves, but they may convey sensitive information about end users' opinions and tastes, which can be misused. On the other hand, labels' and ratings' accountability is fundamental in order to make their authors responsible of what they claim. QUATRO Plus raises other issues surrounding privacy protection and accountability. If user-defined labels and ratings are public (i.e., accessible by everyone, even by people who are not

members of the QSN), the whole Web community can benefit of them. However, this may result in an inappropriate disclosure of private information.

Based on this, when specifying a label or a rating, QSN members are given various protection options:

- the label/rating is private—it can be seen only by the QSN member who created it;
- the label/rating is visible only to the direct contacts of the QSN member who created it;
- the label/rating is visible only to the group members of the QSN member who created it;
- the label/rating is visible to any QSN member;
- the label/rating is public—also non QSN members can access it.

When aggregated, labels and ratings no longer carry information concerning their authors, so they do not explicitly disclose private information. Nonetheless, QSN administrators are given the possibility of setting which labels and ratings must be taken into account (i.e. any label, only public labels, etc.) and whether the aggregate view is a service available only to QSN members or also to non QSN members.

4.3 Web Access Personalisation

As mentioned earlier, we plan to use the aggregate view of labels and ratings as a basis to enforce Web access personalisation. In order to achieve this, QSN members will be given the ability of specifying *user preferences* denoting the action to be performed when detecting a resource associated with a given set of descriptors, having a given trustworthiness. The supported actions are *block* and *notify*. In case of a block action, access to the requested resource is denied, and the user agent returns the notification message. In case of a notify action, access is granted, and the user agent returns the notification message. In both cases, the end users will be given the possibility of verifying why a given action has been performed by displaying (a) the content of the labels associated with the accessed resource, (b) the corresponding aggregate view, and (c) the user preferences which have been satisfied. Note however that, as far as single labels are concerned, the end user will be able to display only the ones he/she is authorised to see, based on the disclosure policies specified by their authors (see Sect. 4.2).

The language and the tools to be used for user preference specification and enforcement are currently one of the issues we are investigating. A possible option is to use a rule language, such as N3 Rules [9,10], and a reasoner (e.g., Cwm[11]). However, languages such as N3 Rules have an expressivity which is far higher than the one required by our user preferences, and they are not necessarily the best solution, in terms of efficiency. For this reason, an alternative option is to develop a user preference language which could be efficiently processed by a piece of software designed specifically for this purpose.

[11] See http://www.w3.org/2000/10/swap/doc/cwm.html.

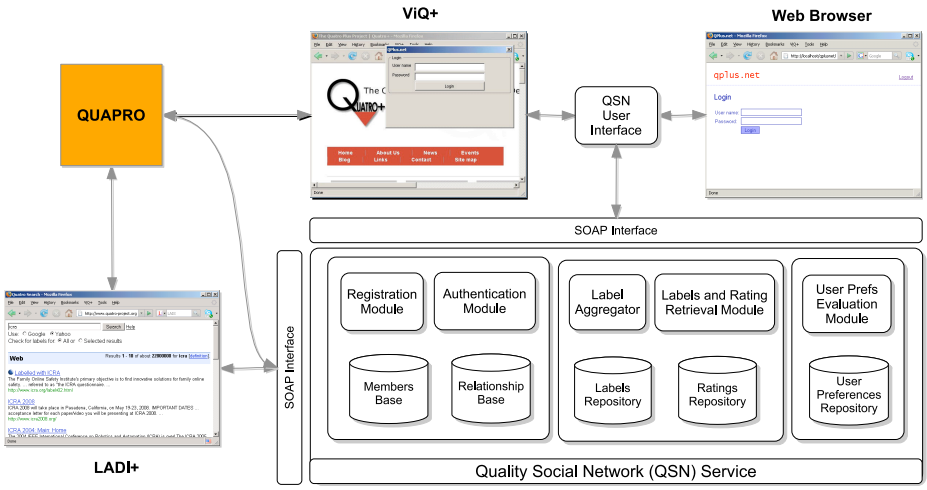

Fig. 6. QSN architecture

4.4 QSN Architecture

The QSN architecture, depicted by Fig. 6, consists of a set of repositories (the member base, which stores members' data and relationships, the user preferences repository, the labels and ratings repositories), storing all QSN data, and of a set of modules in charge of carrying out the supported services (members' registration and authentication, user preferences evaluation, labels and ratings retrieval and aggregation).

In particular, the labels and ratings retrieval module will be in charge of returning the set of labels and ratings satisfying a query, which may concern all the labels and ratings associated with a resource, all the labels and ratings specified by a given member, etc. Such a service will be used by the label aggregator, which is in charge of aggregating and evaluating labels. In turn, the service provided by the label aggregator will be used by the user preferences evaluation module, which will verify whether a given resource satisfies one or more of the user preferences specified by the end user, and then returns the action to be performed by the user agent (ViQ+).

All QSN services will be accessed by the other components of the QUATRO Plus platform through SOAP interfaces. For example, LADI+ will use the results of the statistical analyses of labels and ratings to associate trust values of any available descriptors with search results whilst QUAPRO+ can return the aggregated data regarding the users' ratings on labels applied to a given resource when returning descriptions of it.

The QSN user interface (QUI) will allow end users to register/authenticate on the network, manage their personal data and relationships, specify user-defined labels and ratings, and browse them as a full list or as an aggregate view. As far

as the specification of user-defined labels and ratings is concerned, this task will be performed either by an annotation tool, developed specifically for the QSN, or by the Label Management Environment (LME), which end users will be able to transparently access, directly from the QUI. Finally, the QUI can be accessed by QSN members either from ViQ+ or directly through the browser.

5 Related Work

The idea of annotating Web content is not new. It has been proposed as a means of allowing content consumers to easily identify the content that best fits their needs, by means of using established vocabularies to describe the latter.

Such annotations might relate a variety of content aspects, such as language or languages the content is available in, thematic categorisation, suitability for certain devices, suitability for certain age groups, etc. Web content annotation has many analogies that precede the Web and Web content, such as book and film ratings or food and consumer goods labelling for health and safety reasons.

5.1 Visible Trustmarks

Trustmarks of varying kinds have been in existence for as long as people have communicated. The established online form is a logo on a Web page that, when clicked, returns a certificate delivered from the trustmark operator's system confirming that the trustmark is genuine. This has a number of problems however:

- The user must manually follow a link, check that the certificate is genuine and still valid, and so on.
- The website must permit the content provider to add the logo linking back to trustmark operator, which is not always the case.
- Trustmarks are, typically, only visible on a specific page but refer to the whole site, so they can be missed when following search results or external links.
- The trustmarks are invisible to search engines and linked data systems.

These are precisely the problems that the QUATRO Plus project sets out to address. The 'click to verify' action, and all the authentication, can be done automatically. The result can then be presented to the user through the browser chrome (or some other method independent of the Web page) while the data is also made available in a machine-readable format.

5.2 Semantic Web Approaches to Trust

A great deal of work has been done to enable the assessment of the quality, relevance and trustworthiness of online resources using Semantic Web technologies. Mention has already been made of the WIQA Policy Framework [8]. In that paper, Bizer and Cyganiak identify three metrics for assessing quality: content-based (i.e., analyzing the information content itself), context-based (the available

metadata) and ratings-based. The QUATRO Plus / POWDER approach is a potential source of both context and ratings-based data. A DR may provide any kind of description of one or more resources, including ratings and details such as who created the resource(s) and when, whether a document was created by a company with a commercial interest and so on. Crucially, such descriptions always carry metadata about themselves so that trustworthiness of the context and ratings-based metrics can themselves be assessed.

In [25], Heath and Motta found that in their particular field of study (travel recommendations), *experience* and *affinity* were the key terms for building a system that personalised search results. Since recommendations about travel destinations is inherently subjective, the wisdom of the crowds (mediated through semantic relationships) is entirely appropriate. QUATRO Plus has the potential to contribute additional information to the system such as awards given to restaurants by professional food critics, clean beach awards given by environmental health experts and so on.

In both these cases, QUATRO Plus offers reliable data of known provenance that complements the approach taken.

5.3 Platform for Internet Content Selection (PICS)

One of the W3C's earliest efforts at content labelling was the *Platform for Internet Content Selection* (PICS) [26], addressing some of the issues with using HTTP headers and HTML meta elements to annotate Web content.

PICS specified a format for bundling content annotations along with scoping and attribution information. Scoping amounted to coupling annotations with a URL prefix, including the ability to specify how more specific prefixes override more generic ones. Although this mechanism allowed for the mass-annotation of websites, it was far too restrictive as URL prefixes cannot capture very commonly encountered groupings. Imagine, for example, a site where all pages under http://www.red.example.com/ share features that pages under http://www.green.example.com/ do not.

An important feature of PICS was the *rating system*, a unique identifier for a vocabulary of rating terms. A client application still had to know how to interpret the terms in the vocabulary, but the rating-system mechanism avoided the danger of misinterpreted annotations.

Finally, clients requested PICS documents through a specific HTTP request-for-labels made to either the content provider's Web server or third-party services known as *label bureaux*. This mechanism allowed for independent annotation services, so that content consumers can decide which annotation provider to trust. Client-side trust policies could also refer further attribution information (such as validity dates) included in PICS documents. The major drawbacks of the HTTP request-for-labels mechanism, however, was the requirement for an extension to the HTTP protocol to handle the PICS-specific HTTP request for labels, and that it did not provide for discovery mechanisms so that user agents had to know in advance which label bureaux to ask.

6 Conclusions and Future Work

The linear model of content annotation that posits a label at one end of a chain and a user agent/gateway at the other was developed in the 1990s and promoted particularly by child protection organisations. Experience shows that such a system, for all its political attractions, is very unlikely to gain widespread adoption. In contrast, social networking and shared tags have been phenomenally successful. The QUATRO Plus project and its forerunner (simply called QUATRO) have tried to marry these two facts together as the fundamental point remains: some people really do know more than others about given subjects. In some situations, experience and knowledge must be allowed to counterbalance public perceptions that may or may not be well-informed. By helping to develop a technical platform to support its aims, one that fits in with much broader efforts to develop the Semantic Web; by providing a suitable software infrastructure and by encouraging the publication of interoperable data, complete with an assessment of trust, we believe that balance can be realised.

Any system needs ongoing technical development and maintenance, together with an infrastructure to support it. Alongside that is the need for dissemination and community building, The latter drives the former and it is the building of that community to which the QUATRO Plus partners are now turning their attention.

Acknowledgements QUATRO Plus is co-funded by the European Union's Safer Internet Programme (SIP-2006-211001) and includes partners from industry, academia and the non-profit sector.

References

1. Archer, P., Smith, K., Perego, A.: Protocol for Web description resources (POWDER): Description Resources. W3C Working Draft (April 2009) http://www.w3.org/TR/powder-dr/.
2. Archer, P., Perego, A., Smith, K.: Protocol for Web description resources (POWDER): Grouping of Resources. W3C Working Draft (April 2009) http://www.w3.org/TR/powder-grouping/.
3. Plale, B., Miles, S., Goble, C., Missier, P., Barga, R., Simmhan, Y., Futrelle, J., McGrath, R., Myers, J., Paulson, P., Bowers, S., Ludaescher, B., Kwasnikowska, N., den Bussche, J.V., Ellkvist, T., Freire, J., Groth, P.: The Open Provenance Model (v1.01). Technical Report 16148, University of Southampton, School of Electronics and Computer Science (July 2008) http://eprints.ecs.soton.ac.uk/16148/.
4. Nottingham, M.: Web linking. IETF Internet-Draft, Internet Engineering Task Force (April 2009) http://www.ietf.org/internet-drafts/draft-nottingham-http-link-header-05.txt.
5. Nottingham, M., Sayre, R.: The Atom syndication format. IETF RFC 4287, Internet Engineering Task Force (December 2005) http://www.ietf.org/rfc/rfc4287.txt.
6. Adida, B., Birbeck, M.: RDFa primer: Bridging the human and data Webs. W3C Working Group Note, W3C (October 2008)

7. Carroll, J.J., Bizer, C., Hayes, P.J., Stickler, P.: Named graphs. Journal of Web Semantics **3**(4) (December 2005) 247–267
8. Bizer, C., Cyganiak, R.: Quality-driven information filtering using the WIQA policy framework. Journal of Web Semantics **7**(1) (January 2009) 1–10
9. Berners-Lee, T., Connolly, D.: Notation3 (N3): A readable RDF syntax. W3C Team Submission (January 2008) http://www.w3.org/TeamSubmission/n3/.
10. Berners-Lee, T., Connolly, D., Kagal, L., Scharf, Y., Hendler, J.: N3Logic: A logical framework for the World Wide Web. Theory and Practice of Logic Programming **8**(3) (May 2008) 249–269
11. Konstantopoulos, S., Archer, P.: Protocol for Web description resources (POWDER): Formal semantics. W3C Working Draft (April 2009) http://www.w3.org/TR/powder-formal/.
12. Clark, J.: XSL Transformations (XSLT) — Version 1.0. W3C Recommendation, W3C (November 1999)
13. Connolly, D.: Gleaning resource descriptions from dialects of languages (GRDDL). W3C Recommendation, W3C (September 2007) http://www.w3.org/TR/grddl/.
14. Brickley, D., Miller, L.: FOAF Vocabulary Specification v0.91. Namespace Document (November 2007)
15. DCMI Usage Board: DCMI metadata terms. DCMI Recommendation, Dublin Core Metadata Initiative (January 2008)
16. Booth, D., Liu, C.K.: Web Services Description Language (WSDL) — Version 2.0 Part 0: Primer. W3C Recommendation, W3C (June 2007) http://www.w3.org/TR/wsdl20-primer/.
17. Carminati, B., Ferrari, E., Perego, A.: Combining social networks and Semantic Web technologies for personalizing Web access. In: CollaborateCom 2008. (November 2008) http://www.dicom.uninsubria.it/dawsec/pubs/collaboratecom2008.pdf.
18. Jøsang, A., Ismail, R., Boyd, C.: A survey of trust and reputation systems for online service provision. Decision Support Systems **43**(2) (2007) 618–644
19. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The Eigentrust algorithm for reputation management in P2P networks. In: WWW 2003. (2003) 640–651
20. Xiong, L., Liu, L.: PeerTrust: Supporting reputation-based trust for peer-to-peer electronic communities. IEEE Transactions on Knowledge and Data Engineering **16**(7) (2004) 843–857
21. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge & Data Engineering **17**(6) (June 2005) 734–749
22. Sen, S., Lam, S.K., Rashid, A.M., Cosley, D., Frankowski, D., Osterhouse, J., Harper, F.M., Riedl, J.: tagging, communities, vocabulary, evolution. In: CSCW 2006. (2006) 181–190
23. Golbeck, J.A.: Generating predictive movie recommendations from trust in social networks. In: iTrust 2006. (2006) 93–104
24. Avesani, P., Massa, P., Tiella, R.: A trust-enhanced recommender system application: Moleskiing. In: ACM SAC 2005. (2005) 1589–1593
25. Heath, T., Motta, E.: Personalizing relevance on the Semantic Web through trusted recommendations from a social network. In: ESWC'06 SWP Workshop. (2006)
26. Resnick, P., Miller, J.: PICS: Internet access controls without censorship. Communications of the ACM **39**(10) (1996) 87–93

Assessing Trust: Contextual Accountability

Matthew Rowe and Jonathan Butters

OAK Group, Department of Computer Science, University of Sheffield, Regent Court, 211 Portobello Street, Sheffield, United Kingdom

{m.rowe,j.butters}@dcs.shef.ac.uk

Abstract. The need to assess the trustworthiness of a piece of information has become a prevalent issue due to web vandalism and the openness of platforms such as wikis. In this paper we introduce the notion of contextual accountability: Holding the author of a given statement accountable by deriving a path from the statement to the author's credentials. We present our own trust vector model as a combination of the assessment of these credentials covering two aspects; knowledgeability and social acceptance. This model is put to the test using two datasets and the results are presented and discussed.

1 Introduction

With the abundance of user generated content on the web comes the need to filter or weight the validity of statements and factoids. This ability is innate in humans but is completely lacking in agents and software that blindly rely on linked data and triples as hard fact. There are many types of users that contribute information to online sources, and these can generally fall into positive contributors and negative contributors. Negative contributors can be subdivided into three types: Troll - a knowledgeable user who posts negatively biased posts to create angst amongst the community, Spammer - a non-knowledgeable user who floods the community with unrelated posts usually offering a product or service, and Truth Engineer - a highly knowledgeable user who generally contributes to a very narrow domain, often trying to manipulate the community's point of view to better align with some ideal. It is therefore obvious that different levels of trust should be given to different statements made by authors based on the their areas of expertise and their posting habits within the sub ject area. In this paper we propose a methodology for assessing the level of trust given to statements made in a particular context by estimating the level of knowledge the author has in this field and how respected this author is by community members.

Wikipedia is a prime target for Trolls and Truth Engineers. The Wikipedia article regarding Cypriot football team AC Omonia included bogus information regarding a fan club lovingly called 'The Zany Ones', who wore 'shoes on their heads', had a song 'about potato' and 'kept their season tickets in the oven for safekeeping'. Attention was brought to these edits when they were inadvertently

taken as fact and printed in a national UK tabloid on the 18th September 2008[1]. The reliance on knowledge sources such as Wikipedia has also forced Wikipedia to rethink their edit policy for pages about living people and companies[2] following a spate of malicious edits on such pages. The proposed changes to the edit policy will see trusted community members vetting edits prior to their inclusion in the page.

In this paper we present our method for deriving trustworthiness of a person given a context, relying on holding a person accountable for their actions. Semantic Web technologies and formalisations allow graphs to be created containing links between individual semantic resources. Paths through such a graph space provide a technique whereby a given statement can be traced back to the statement author, thus holding that person accountable. We define accountability as the possibility to generate this path structure, which is only achievable through a semantic graph space using linked data across multiple data sources and knowledge bases. The bill of rights for web users[3] gives grounding to this theory by treating web users not simply as consumers but as citizens, making them responsible for their own actions. This is the stage that we believe the web has now reached, where data sources must be assessed for validity by assessing the statement authors and contributors.

Deriving a value of trustworthiness for a person given a context requires the assessment of known information that can support this person's claim as a trusted entity. In this paper we divide this information into two distinct aspects; knowledge and social. The former referring to formal information accredited to the individual such as publications or reports, essentially providing validation that this person's expertise is evident within the context in question. The latter refers to the community acceptance socially attributed to the person; in this case it can be message board posts, or blog posts critiqued by the community. We combine each aspect to provide a vector representation of the person's overall trustworthiness, depending on the position in the vector space, it is possible to derive a classification for the person in question and whether they should be trusted or not, given the context.

This paper is structured as follows: Section two presents an approach to model accountability paths through a semantic graph space from a given statement through to the credentials of the statement author. Section three describes the metrics used to derive trustworthiness from both the social and knowledge aspects of a statement author's credentials, and how these measures are combined to produce a vector representation. Section four describes the experiments conducted to demonstrate the effectiveness of our approach, and the discussion of the obtained results. Section five compares state of the art trust derivation techniques with our approach, and section six describes the conclusions drawn from this work and discusses planned future work.

---

[1] http://www.thespoiler.co.uk/index.php/2008/09/19/daily-mirror-football-journalist-merked-by-wikipedia

[2] http://technology.timesonline.co.uk/tol/news/tech_and_web/the_web/article5593986.ece

[3] http://opensocialweb.org/2007/09/05/bill-of-rights/

2 The Semantics of Accountability

In order to assess for trust we rely on the existence of linked data to derive a path through the web from the a given statement through to credentials attributed to the statement author. We define a statement as a loose abstract notion of a piece of knowledge added to a knowledge base (e.g. OWL Full is more expressive than OWL DL). In essence we wish to take a statement and derive a trust metric for this statement given two properties: The author, and the context. The credentials we define cover two different aspects: knowledge and social. The former deals with formal work attributed to the statement author, and the latter focuses on the community recognition attributed to the statement author. In each case, the goal is to assess each aspect with respect to the context of the statement in question.

2.1 Gathering a semantic representation

To contextualise this, imagine a person updates the Semantic Web Wikipedia page. In order to assess the trustworthiness of this update we assess the credentials associated with the author, holding them accountable for this update. Both the social and knowledge aspects are combined to derive a vector interpretation of the author's trustworthiness given the context of the statement, which in this case is about the Semantic Web. The derivation of this vector is explained in detail in the following section.

At a low level, holding a statement author accountable requires the derivation of a path through the web from the statement through to the credentials attached to the statement author. Linked data and Semantic Web formalisations enable such paths to be derived using the intrinsic graph structure of semantic formalisations as we will now explain. We begin by taking a statement to be analysed, given the nature of knowledge bases (e.g. Wikipedia, Freebase, etc), each statement can be related to an author representation within that platform, e.g. A profile page about the author. At this stage we assume that a semantic representation of this author exists as RDF according to the FOAF specification [2], however we must find this information. Therefore extracting this semantic representation is carried out using one of the three methods:

1. Query the Semantic Web for explicit representation: The author's handle or username is submitted to a Semantic Web entry point such as Watson[4] or Swoogle[5], and the relevant FOAF file is extracted.
2. Query the wider web for semantic representation: The author's handle or username is submitted to a web search engine such as Google[6]. The most relevant page is assessed for the existence of lightweight semantics such as Microformats[5] or RDFa [4], and explicitly linked FOAF files.

[4] http://watson.kmi.open.ac.uk/WatsonWUI
[5] http://swoogle.umbc.edu
[6] http://www.google.com

3. Implicit semantics within the existing profile page: Several platforms now support the use of RDFa and Microformats within XHTML. Given this use of implicit semantics, the profile page is parsed to derive credentials relating to the knowledge and/or social aspects, or linked data representations where such information can be found (ie. Hyperlink to a FOAF file).

2.2 Deriving Credentials

Given that we now have a semantic representation describing the statement author, we assume that this formalisation contains the credentials of each aspect or a linked data representation of such information. Credential information describing the knowledge aspect is formalised using the Bibtex specification[7], where each paper is an instance of `bibtex:Entry` or one of it's subclasses, and paper details are defined using `bibtex:hasTitle` for the title, `bibtex:hasKeywords` for keywords, and `bibtex:hasBookTitle` for the title of the publication. The context of the statement is analysed against these properties, given that we do not wish to assess all publications attributed to the author, simply the publications within the context of the statement, thus deriving a subset of the original publications set.

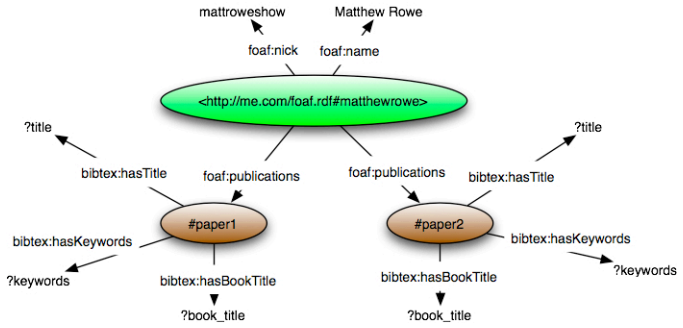

Fig. 1. Statement author linked to publications as instances of bibtex:Entry

Credential information related to the social aspect is derived using linked data within the FOAF file. The instance of `foaf:Person` describing the statement author is linked to an instance of `sioc:User` using the `sioc:account_of property`. As SIOC [1] is purposely designed to define online community structures, a weblog or forum is attributed to the statement author using the `sioc:owner_of` property property with the `sioc:User` as the domain and `sioc:Container` as the range. Any blog or forum posts are then made within this container, where each post is an instance of `sioc:Post`. Our intuition at this stage is that the blog attributed to the statement author will contain numerous posts about a variety

[7] http://zeitkunst.org/bibtex/0.1/bibtex.owl

of subjects, therefore in a similar manner to the knowledge aspect we derive a subset of the total set of blog posts containing instances of `sioc:Post` related to the statement context. This is fascilitated by the use of the sioc:topic property to denote tags assigned to the blog posts, we compare these properties against the context and filter out all instance of `sioc:Post` that do not match. Figure 2 demonstrates the RDF graph structure linking an instance of `foaf:Person` due several blog posts as instances of `sioc:Post`.

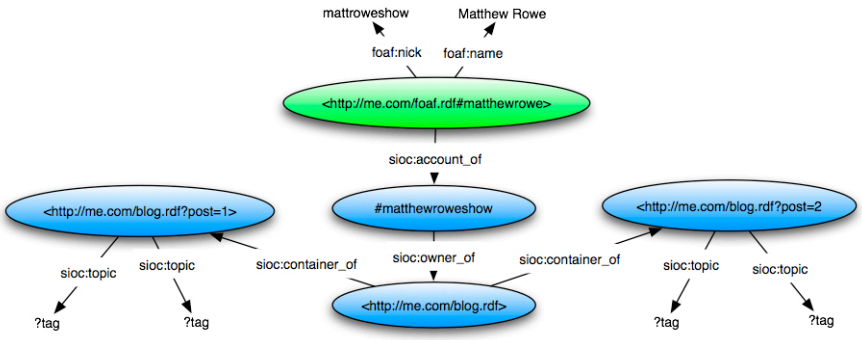

Fig. 2. Statement author linked to his/her blog posts as instances of sioc:Post

Once we have gathered the credential information spanning both the knowledge and social aspects, trust metrics must be derived using this information. We now discuss our proposed metrics with examaples demonstrating the derivation technique.

3 Deriving Contextual Trust

We define two orthogonal vectors; the social aspect, and the knowledge aspect, of which the vector addition results in the trust vector. The social aspect quantifies how respected the author's point of view is in a given domain by taking into consideration positive feedback and recent activity levels, similarity the knowledge aspect assesses the level of trust associated to scientific statements the author has made.

3.1 Social Aspect

We define the social trustworthyness of a given person within a certain knowledge topic as follows. Let p be the person in question, c is the topic of knowledge covered by the statement, P' is the set of posts (on a discussion board or blog) made by p about the topic c, F is the set of comments or feedback elements attributed to a given post. We further define the function $acceptance(f)$ as the classification of a feedback element describing acceptance in relation to the post

or not, this will return +1 if the feedback is positive, -1 if the feedback is negative and 0 if the feedback is neutral. $\left|P^{'}\right|$ denotes the number of posts made by that person about that topic, which we sqaure in order to increase the value of the posting more (different values were experimented with, squaring appeared to produce the optimum value). Therefore the social trustworthyness of a person given a knowledge context is as follows:

$$S(p,c) = \frac{\sum_{p' \in P'_{p,c}} (\sum_{f \in F_{p'}} acceptance(f)) \cdot (\left|P^{'}\right|)^2}{timeperiod}$$

3.2 Knowledge Aspect

We define the knowledge trustworthyness of a given person within a certain knowledge topic as follows. Let p be the person in question, c is the topic of knowledge covered by the statement. Q is the set of papers of which p is an author. With regards to the the knowledge topic c we wish to analyse the set of corresponding papers, we therefore define $R = Q \mid c$ as the set of papers within the knowlege topic, or context. We define a function $citations(r)$ that returns the number of positive citations the paper r has received: Each citation is assessed; positive citations receive +1, negative citations receive -1 and neutral citations receive 0. Using these defintions the knowledge trustworthyness of a person given a knowledge context is as follows:

$$K(p,c) = \frac{\sum_{r \in R} citations(r) \cdot \left(\sum_{y=1}^{x} \frac{|R_y|}{y}\right)^2}{timeperiod}$$

The formula takes in to account the recent activity of the person in question: The number of citations received in a certain year are divided by how many years ago the citations were made. This is denoted by $\frac{|R_y|}{y}$ where y ranges from 1 year ago to x number of years ago. Such a weighting is valid because within the Semantic Web domain, and indeed in other areas of web science, techniques and work moves extremely quickly. Therefore a more heavily cited recent piece of work should carry more weight than an older piece of work with the same number of citations.

3.3 Weighting Knowledge Aspects

As trust is a relative quantifier, the level of trustworthiness can only be calculated between the group of people who have made contributions to a particular resource, to this end, once the knowledge and and social scores for each person has been calculated for a particular context, each score must be normalised by the the magnitude of the maximum score in the sample set for each aspect. Trustworthiness is then represented by the resultant vector given by:

$$\overrightarrow{T} = S(p,c)\hat{S} + K(p,c)\hat{K}$$

where $\hat{S}$ and $\hat{K}$ are two orthogonal dimensions which form our vector space.

The resultant Trustworthiness vector $\overrightarrow{T}$ for each person will have a maximum potential magnitude of $\sqrt{2}$ and may lie in either of the four quadrants. By observing the location of each vector it is possible to assess the type of poster the user is. Individuals with higher positive social aspect scores tend to be more of a 'familiar name in the community' who's level of trust stems from the notion that they are observers (i.e. visible lurkers) within the context whose contribution to topic discussions are generally positive. Individuals with higher Knowledge aspect scores tend to be contributors to the context whose work tends towards to the state of the art.

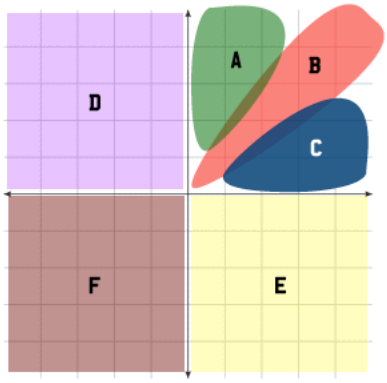

Fig. 3. Vector space representation of trustworthiness classifications. The x axis represents the knowledge score and the y axis represents the social score.

As it is possible for an individual to attain negative values for social and knowledge aspect vectors, it is possible for an individual to attain for instance, a negative value for the social aspect and a positive value for the knowledge aspect. This is indicative of a person who objects to points raised in conversations (such as challenging people's points of view with forum or blog posts), but who's work is original and is therefore cited.

Our intuition behind the design of the trust metrics and the resultant vector model is to allow classification of statement authors, and therefore reach a decision whether they should be trusted or not. If one considers figure 3, any person whose trust vector lies within the areas marked A, B or C can be considered a trustworthy person, with larger vector lengths representing a greater level of trust. However, should a person have a trust vector residing in the area marked D that classifies the person as having a good level of social trustworthiness, yet

their formal publications are not supported by the community (e.g. Einstein prior to his theory of relativity being proved by Ellington). Similarly, should the trust vector lie in the area marked E one could consider this person to be formally accepted as knowledgeable, yet socially their ideas are not accepted. (e.g. A truth engineer; CIA, geeks, nerds, etc). In either case, where the vector is in area D or E it is not certain whether this person should be trusted or not.

Finally, if the trust vector lies in the area marked F, it is fair to consider this person unstrustworthy due to their exclusion by the community socially and formally in their work. Our belief is that trolls and spammers would reside in this area given that their ideas are, in general, not supported by the community and their social trustworthiness is negative due to their motivations of using the web being for maliscious purposes.

4 Experiments

In order to assess the validity of our trust metrics we designed two experiments covering two separate domains of expertise; the Semantic Web and HCI (human computer interaction). In each domain we assumed that a wikipedia page had been edited and that statements within that page were linked to a set of people. In each case we derived the list of people based on conference proceedings and workshop organising committee lists to find those people who have both a blog and a list of publications. We then derived trust measures manually, checking positive and negative feedback and citations for the social and knowledge aspects respectively. The results were then collated and analysed.

4.1 Datasets

Two datasets were gathered for the experiments, where each dataset contained a list of people within a separate domain of expertise. For the Semantic Web dataset we obtained the list of potential Semantic Web statement authors from the linked data on the web workshop[8], social data on the web workshop[9] and european Semantic Web conference[10] participants lists. From these lists we then selected people with both papers and a blog, making sure that we chose a diverse range of experience and duration within the domain. A similar action was performed for the separate domain Human Computer Interaction (HCI), whereby authors of prominent papers were sampled.

4.2 Results

The results from the Semantic Web dataset demonstrate how for the people in the dataset, the majority carried a strong emphasis on blogging their work which

[8] http://events.linkeddata.org/ldow2009/
[9] http://sdow2008.semanticweb.org/
[10] http://www.eswc2009.org/

Name	Knowledge Aspect	Social Aspect	Relative Trustworthiness
Person A	0.085239676	0.251591844	0.265637592
Person B	0.393860438	0.683222914	0.788618789
Person C	1	0.007884645	1.000031083
Person D	0.120373117	1	1.007218788
Person E	0	0.008778798	0.008778798
Person F	0.12748067	0.363994004	0.385672084
Person G	0.121361371	0.111505714	0.164809304
Person H	0.090343433	0.070168176	0.114391909
Person I	0.15023741	0.026336395	0.152528309
Person J	0.27146477	0.011150571	0.271693682

Table 1. Trust statistics from the Semantic Web dataset

Name	Knowledge Aspect	Social Aspect	Relative Trustworthiness
Person K	1	0	1
Person L	0.22574209	0.007674504	0.225872507
Person M	0.76437217	0	0.76437217
Person N	0.079107224	1	1.003124096
Person O	0.219654267	0	0.219654267
Person P	0.120053261	0.034494332	0.124910545
Person Q	0.807120783	0	0.807120783
Person R	0.013191016	0.049538713	0.051264871

Table 2. Trust statistics from the human computer interaction dataset

Fig. 4. Vector space representation of Semantic Web dataset (left) and the HCI dataset (right). The x axis represents the knowledge score and the x axis represents the social score.

was in turn positively rated by the community of readers in the blogosphere. There are two evident extreme cases with respect to each aspect: Person D for the social aspect, and person C for the knowledge aspect. As the former blogs regularly it is no surprise that he sets the precedent for maximising trust value. So too in the case of person C, who has numerous publications all of which are heavily cited in the Semantic Web community. Figure 4 throws up some interesting questions though. Using our combination of each aspect measure into a single vector yields the derivation that person D is the most trustworthy within the Semantic Web dataset, following closely by person C.

Another interesting feature of these results is the correlation between the vectors of person A and person F. Although the former vector has less magnitude than the former, it falls along the same angle suggesting a correlation of trustworthiness with a similar spread between blog posts and publications.

Although the formula takes steps to mitigate the disparity between the level of trust extremely large numbers of citations and more average numbers of citations achieve, it can be seen that the Semantic Web results appear skewed on the knowledge aspect, this is due to the fact that one person was a prolific and heavily cited author of papers resulting in an overshadowing effect on the publishing counts of others. The HCI dataset results vary from the Semantic Web results due to the fact that many of the blogs that the HCI people maintain do not achieve levels of comments or feedback the Semantic Web blogs do, and therefore social trustworthiness in the HCI context is lower. However, as we define trustworthiness as the vector addition of two orthogonal aspects we maintain the ability to rank people according to their measured knowledgeability. The main reason for the higher number of people in the Semantic Web domain with social trust is most likely due to the Semantic Web experts' use of the blogosphere as a medium for publishing and sharing content as well as providing a quick way to expose ideas and discuss them. This increases other people's awareness of them as a positive influence, which will in turn increase their social trust score.

If we contrust figure 4 and figure 3, the positioning of the participants all lie in the upper right quadrant of the vector space. When analysed for negative feedback and citations, it became apparent that the majority were positive, and only a few negatives for each person. Our belief is that given a larger experiment incorporating a more controversial, and disputed domain of work, criticisms and negative comments would play a major role in deciphering trust with respect to a person and context.

5 State of the Art

In the Semantic Web community, work has been performed to investigate the network of trust necessary for Semantic Webs to function. Work by [3] proposes deriving trust values from an existing network too allow application and similar services that may wish to interface with the network to have access to an average trust value for the network. Unfortunately, altough each edge in the graph is denoted by a trust value, it is unclear how this value is derived. The notion

of trust is dealt with in work by [8] through the use of bilattice frameworks to compute trust associated with a given statement, similar to the goal of our work. [8] differs from this paper however, by utilising multiple trust levels depending on the data source hosting the statement. Semantic web services require trust to be estabished between the service requestor and provider, in such a problem trust information must be passed between the two parties. [6] describes such trust policies that when passed between the parties, must be met in order for trust to be achieved. Policies commonly contain a list of criteria in the form of logic statements. The majority of trust based research conducted within the Semantic Web utilises black box techniques to derive a trust measure for a given statement or person.

External to web semantics, communities such as security and privacy have provided approaches for modelling trust. With regards to the actual vector model demonstrated in this paper, work presented in [7] by Rat et al is comparable by providing a vector representation of trust associated with a given person within a given context as a combination of an experience aspect, knowledge aspect and cumulative effect aspect. Both our work and work by Ray et al utilises the context as an essential feature of trust derivation, however our work differs by computing relative trust scores among a group of statement authors. A very similar approach to our work is described by Kim et al in [9] by deriving the degree of trust for a given person from the affiliation and expertise the person has with the context. The model used by Kim et al is similar to our work, by creating a path from the statement author, which in the case of [9] is a reviewer, to their create content and the relevant replies.

6 Conclusions and Future Work

In this paper we have presented an approach to hold the author of a statement accountable, and provide a path of accountability from the statement to the author credentials through a semantic graph space. Our interpretation of trust as a vector allows the portable model of trust to be computed in a number of domains. As we have presented in this paper with the Semantic Web and HCI datasets. The exclusion of an evaluation of the derived trust metrics was largely due to the unavailability of sufficient evaluation participants at the time of conducting the experiments. It is vital for evaluation participants to have an understanding of the field in question – Semantic Web or HCI – in order to make an informed decision. We plan to investigate this further in the future.

Limitations meant that we were only able to manually examine a pre-chosen selection of people for the experiments resulting in our test individuals, scoring positive results. Future work will investigate experiments using additional datasets known to contain negative contributors, effectively producing a classification of the type of negative web user.

During the analysis of the results a third aspect of trust became more apparent. The notion that a small set of people may trust one another more because they are co-located. Although this aspect of trust is important in working rela-

tionships, where a higher level of direct accountability means that a person will generally assign a high degree of trust to those they work with, the requirement of a personal point of view makes the application of this trust metric to general statements impossible, unless of course the overall level of trust attained were only applicable to a single person's point of view, in this case, a third trust vector can simply be added. One addition to this separate aspect could include the notion of vouching. The chain of reliability in a trust network would denote that person A trusts person B, and therefore vouches for them. However, the level of trust assigned to person B could only be the maximum trust measure that person A currently has. Should that measure be low, then person B would only receive a weak trust measure from person A.

References

1. Uldis Bojars, Alexandre Passant, Richard Cyganiak, and John Breslin. Weaving sioc into the web of linked data. In *Proceedings of the WWW 2008 Workshop Linked Data on the Web (LDOW2008), Beijing, China*, Apr 2008.
2. Dan Brickley and Libby Miller. FOAF vocabulary specification. Technical report, FOAF project, May 2007.
3. Jennifer Golbeck, Bijan Parsia, and James Hendler. Trust networks on the semantic web. pages 238–249. 2003.
4. W3C Working Group. Rdfa primer: Bridging the human and data webs, October 2008.
5. R. Khare. Microformats: The next (small) thing on the semantic web? *Internet Computing, IEEE*, 10(1):68–75, 2006.
6. Daniel Olmedilla, Ruben Lara, Axel Polleres, and Holger Lausen. Trust negotiation for semantic web services. In *1st International Workshop on Semantic Web Services and Web Process Composition in conjunction with the 2004 IEEE International Conference on Web Services*, pages 81–95. Springer, 2004.
7. Indrajit Ray, Sudip Chakraborty, and Indrakshi Ray. Vtrust: A trust management system based on a vector model of trust. In *In Lecture Notes in Computer Science*, pages 91–105. Springer, 2005.
8. Simon Schenk. On the semantics of trust and caching in the semantic web. In *ISWC '08: Proceedings of the 7th International Conference on The Semantic Web*. Springer-Verlag, 2008.
9. J. Young Ae Kim Minh-Tam Le Lauw, H.W. Ee-Peng Lim Haifeng Liu Srivastava. Building a web of trust without explicit trust ratings. In *Proceedings from Data Engineering Workshop, 2008. ICDEW 2008. IEEE 24th International Conference on*, pages 531–536, 2008.

Policy-based Access Control

Trust and Privacy on the Social and Semantic Web
(SPOT 2009)

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.

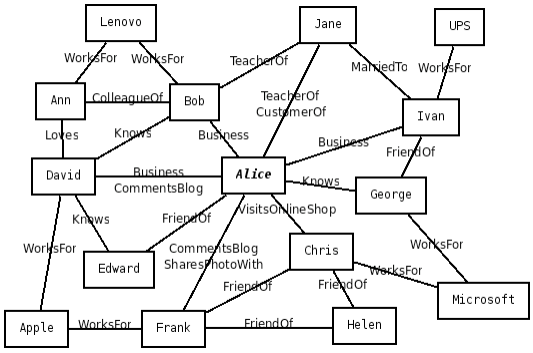

Fig. 1. Alice's Social Network

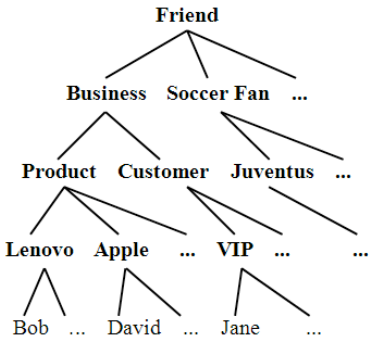

Fig. 2. Alice's Social Ontology

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

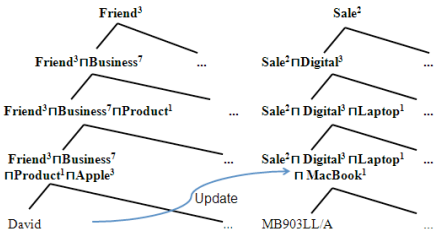

Fig. 4. Permission Assignment on Lightweight Ontologies

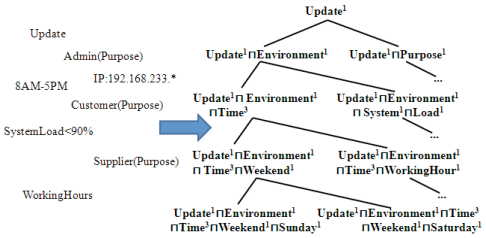

Fig. 5. Scattered Permissions to a Lightweight Ontology

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

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

- $Sale^2 \bot Friend^3$ 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 $Sale^2$.
- $MacBook^1 \bot Lenovo^1$ comes from that '*MacBook is a product of Apple company but not Lenovo.*' This kind of mismatch suggests exactly no access should be assigned.
- $Sale^2 \bot (Soccer^1 \sqcap Fan^2)$ 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,

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

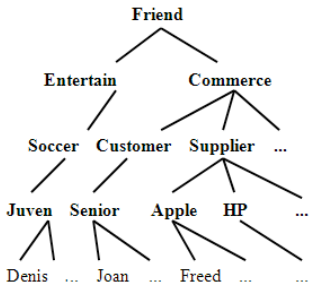

Fig. 6. Bob's Social Ontology

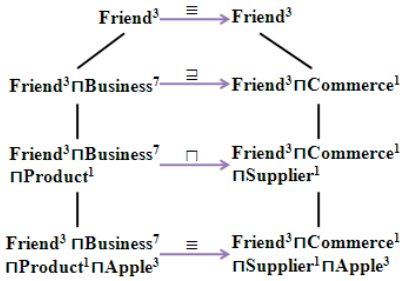

Fig. 7. Ontology Matching for Rule Reuse

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

Using Natural Language Policies for Privacy Control in Social Platforms*

Juri L. De Coi[1], Philipp Kärger[1], Daniel Olmedilla[2], and Sergej Zerr[1]

[1] L3S Research Center & Leibniz University of Hannover, Germany
[2] Telefónica Research & Development, Madrid, Spain

Abstract. The ability of defining privacy preferences in the current social platforms are very restricted. Typically, the user is provided with only some predefined options to select from. In this paper, we present an approach that exploits Semantic Web policies allowing users to control privacy in social web applications. Such policies are general statements that define the behavior of a system. Although Semantic Web policies gained a lot of interest in recent years and policy languages became more and more complex, suitable and easy-to-use solutions for highly dynamic social platforms are still needed. In order to allow *common* users to define policies about the data to be shared, we introduce *natural language* policies. We further present an implementation based on the policy framework Protune that allows users of a collaborative learning platform to restrict access to their learning material to collaborators by means of controlled natural language policies.

1 Introduction

Since the web became a place where people are not only consuming content but creating, publishing and sharing content, it is needed to allow people to exactly define who is allowed to access which part of the content they provide. Unfortunately, there is currently no simple way to restrict access to some content to only a set of trusted parties not previously known or to decide who is allowed to access some part of a profile [1]. In Flickr for instance, you are allowed to define people as friends or family members and, based on those assignments, it can be decided who is allowed to see what picture. Still more fine-grained access control management is lacking. Thus, if both, colleagues and close friends, are in the same group, there is no way to restrict access to a subgroup for a particular picture [1]. Other web applications for sharing data or social platforms do also allow only for similar limited access control features leading to a high potential of leaking private data [2].

Semantic Web policies have gained a lot of interest in the last years and promising approaches show their applicability to real-life applications, be they in the area of web personalization, agent and network control, or access control on the web. For this reason a number of policy languages have been defined in the last years. Nevertheless a major hindrance to widespread adoption of policy languages are their shortcomings in terms of usability: in order to be machine-understandable all of them rely on a formal syntax, which common users find unintuitive and hard to grasp. Therefore, in most such applications, policies are required to be authored by system administrators or specifically trained people.

* The authors' efforts were partly funded by the European Commission in the TENCompetence project (IST-2004-02787) (http://www.tencompetence.org).

In this paper, we present an approach that overcomes both mentioned shortcomings: the limited access control capabilities of existing collaborative systems and the complexity of current policy languages. First, we do not restrict the user to role-based access control on the basis of user groups or similar but we allow users to specify general policies providing fine-grained access control. Second, our approach does not require users to define policies in formal languages but provides an interface for natural language policies.

In the presented approach, we apply Protune and its corresponding policy language. By doing so we additionally benefit from the features of an advanced policy engine such as the generation of natural language text explaining the result of a policy evaluation (e.g., why access was denied) or the support of trust negotiation [3]. Further, we use an adapted version of the controlled natural language ACE to define policies.

The remainder of this paper is structured as follows. In the next section, we identify requirements a general privacy control solution should fulfill. In Section 3 we provide some background information on Semantic Web policies and controlled natural languages. Then, in Section 4 and 5 we detail how policies can be used to control privacy in social web applications. Section 6 introduces our prototype and Section 7 concludes the paper.

2 Motivation and Problem Statement

Preserving privacy in social platforms is a known challenge (see [4] for an information leakage on Facebook or [5] for a similar case on Flickr). Existing social platforms provide only limited means to define access control rights [6]. The problem is that privacy preferences have to be easy to define but still expressive enough to capture all possible cases and combinations of a user's wishes to define who is allowed to access what. An additional challenge is the dynamics of nowadays social networks. We identified the following requirements for a solution aiming at preserving privacy on a social platform.

Adaptive to social network dynamics. Since nowadays social networks change rapidly, any solution should be able to easily adapt itself to changes in the data of requesters.

Fine-grained. Privacy preferences can be arbitrary fine-grained. If a picture shall be shared only with a restricted set of people (maybe not even known in advance), it should be easy to express such requirement.

Extensible. Privacy preferences should be easily extensible: imagine a user uploads a new set of pictures to a social platform and she wants them to be shared with attendees of a conference. The user's current privacy preferences should be easily extensible without requiring all conference attendees to be registered at the social platform.

Natural language interface and feedback. Defining privacy preferences has to remain a simple and straightforward process. Access control decisions should be transparent and well explained to users. Similarly, the specification of privacy preferences has to protect users from collections of check boxes defining which friends is allowed to access which file or from similar complicated policy definitions.

Security mechanisms. Last, but not least, any solution must fulfill basic security and privacy requirements, such as reliability, support to authentication, delegation of rights and evidences (such as credentials and declarations), etc.

3 Background

3.1 Semantic Web Policies

Semantic Web policies are statements defining the behavior of a system. Access control policies define the conditions to be fulfilled in order to get access to a certain resource. Policies are pervasive in all web-related contexts [7] and are needed to protect any system open to the internet. They declaratively specify requirements which must be fulfilled but do not state how they have to be fulfilled, thereby providing a separation between specification and enforcement. Privacy policy frameworks such as the Platform for Privacy Preferences (P3P) or EPAL are exploited to assist users while they are browsing the web and interacting with web services. The early policy frameworks were rather decision support systems than systems for declarative behavior control [7]. Moreover, they were not expressive enough to fully address Semantic Web scenarios involving reasoning, ontology-based knowledge representation, etc. Nowadays, a number of policy languages have been developed in order to address the challenges of the Semantic Web: expressiveness, simplicity, enforceability, scalability, and analyzability [8]. Among the most prominent ones (see [14] for a broad overview of policy languages and their features) we mention KAoS [9], Rei [10], PeerTrust [11], and Protune [12]. In order to provide a well-defined semantics these policy languages are typically based on languages logical formalisms like description logics (e.g., KAoS and Rei) or logic programming (e.g., Cassandra [13], PeerTrust, and Protune). Policy engines are in charge of evaluating policies (i.e., of checking whether they are satisfied or not).

The Policy Framework Protune
In our approach we used the Protune[3] (PRovisional TrUst NEgotiation) framework for policy evaluation and enforcement. Protune [12] aims at combining distributed trust management policies with provisional-style business rules and access control-related actions. Protune features an advanced policy language for policy-driven negotiation and supports distributed credentials management and flexible policy protection mechanisms. The Protune policy framework offers a high flexibility for specifying any kind of policy, referring to external systems from within a policy and providing facilities for increasing user awareness, like automatic generation of natural language explanations of the result of a policy's evaluation.

The Protune policy language is Logic Programming-based and as such a Protune policy has much in common with a Logic Program. An example Protune policy is

$$allow(access(Requester, Resource)) \leftarrow friend(Requester), family\_picture(Resource).$$

states that a requester can access a resource if the requester is a friend and the resource is a family picture. (According to the standard Logic Programming conventions, terms starting with a capital letter refer to variables; that is, *Resource* indicated that the policy applies to every resource.) Adding the following facts $family\_picture('holiday.jpg')$. and $friend('Bob')$. will allow Bob to access the picture `holiday.jpg`. In general, a Protune policy rule has one of the following formats:

$$allow(action) \leftarrow condition_1, \ldots, condition_n.$$

[3] http://policy.L3S.uni-hannover.de/

$$condition \leftarrow condition_1, \ldots, condition_n.$$

(where $n \geq 0$), meaning that the action *action* is allowed (resp. the condition *condition* holds) if all conditions $condition_i$ hold.

A policy may require that, in order to evaluate a request, some actions are performed, that is, some of the $condition_i$ can be actions themselves. For example, rule

$$allow(action_1) \leftarrow action_2.$$

can be read as: $action_1$ *can be* executed if $action_2$ *has been* executed. In a typical scenario access to some resource is allowed only if the requester provides a valid credential (in this case, $action_2$ would be sending a credential). Notice the different semantics of the actions according to the side of the rule they appear in: the execution of actions appearing in the left-hand can be requested by some party, whereas the actions appearing in the right-hand side have to be performed in order to accomplish the request. In order to stress this semantic difference, actions in the left-side of a rule must appear inside the *allow* predicate.

The evaluation of a policy may require to deal with objects which can be modeled as sets of $(attribute, value)$ pairs. The Protune language allows to refer to such properties of an object by means of so-called *complex terms*.

$$identifier.attribute : value$$

For example, a rule stating that only credit cards whose company is VISA are accepted could look like the following.

$$accepted(CreditCard) \leftarrow CreditCard.company :' VISA'.$$

3.2 Controlled Natural Languages

Controlled natural languages (CNLs) are formal languages which have been designed in order not to look like formal languages but to be more user-friendly. As formal languages they are described by a formal grammar each well-formed sentence must comply to. Moreover, they embed disambiguation rules in order to deterministically disambiguate sentences which in full natural language have different readings, as it happens with the following standard example: *Bob looks at the girl on the hill with a telescope.* In full natural language such sentence can have at least three different meanings according to the context it appears in, namely

- *Bob looks with a telescope at the girl who is on the hill.*
- *Bob looks at the girl who is on the hill and has a telescope.*
- *Bob is on the hill, has a telescope and looks at the girl.*

Although the last reading may be considered a bit unlikely, it cannot be excluded. In order to disambiguate such sentences, CNLs usually come along with a set of built-in rules which allow to deterministically select one out of n possible readings. For instance the CNL ACE assumes that all prepositional phrases refer to the predicate, therefore in the example above the action "looking" happens "towards the girl", "on the hill" and "with a telescope", boiling down to the third one of the readings listed above.

Ambiguity resolution takes place on a purely syntactic basis, so that CNLs typically do not differentiate between "God's sake" and "John's love", being both of them built

according to the pattern `[proper name]'s [common noun]`, although in everyday's speech one probably means that it is God to be loved whereas it is John to love.

The ACE controlled natural language

In our approach we use Attempto Controlled English (ACE [15]). ACE is a controlled natural language developed at the University of Zurich with the aim to serve as specification and knowledge representation language on the Semantic Web. ACE is intended for professionals who "need to use formal notations and formal methods, but may not be familiar with them"[4]. The Attempto Parsing Engine (APE) deterministically translates ACE texts into a semantically equivalent internal format (Discourse Representation Structure—DRS) which is easier to further process in an automatic way. ACE and APE come along with a set of tools able to translate DRSs into various other languages (e.g., FOL, PQL, FLUX, RuleML) and a rule format to be used for Courteous Logic Programs [16] and stable model semantics [17]. A prototypical bidirectional translation of ACE into and from OWL DL has been developed as well.

4 Controlled natural language for easy specification of privacy policies

A major hindrance to widespread adoption of policy languages are their shortcomings in terms of usability: in order to be machine-understandable all of them rely on a formal syntax, which common users find unintuitive and hard to grasp. The use of controlled natural languages can improve usability of policy languages. This section describes how we exploit (a subset of) ACE in order to express policies and describe the mapping between ACE policies and the policies written in the Protune language. The full details of the ACE→Protune translation are available in [18]. The following ACE policies are corresponding to the Protune policies introduced in Section 3.1.

- If the requester is a friend and the resource is a family-picture then the requester can access the resource.
- If the company of a credit-card is "VISA" then the credit-card is accepted.

These examples show which features of (ACE and therefore of) the English language can be used in order to define Protune policies: common nouns (like `requester`), adjectives (like `accepted`), verbs (like `access`), genitives (like in `company` *`of`* `a credit-card`) and strings (like `"VISA"`). The following example also shows that proper names (e.g., `Bob`), adjectives in comparative (e.g., `older than`) as well as in superlative form, integers (e.g., `18`) as well as reals, prepositional phrases (e.g., `in "adult-content-folder"`), saxon genitives (e.g., `Bob's`) and relative pronouns (e.g., `everything` *`which`* `is`) are supported too.

> If the requester is older than 18 and she is Bob's friend then she can access everything which is in "adult-content-folder".

As described in Section 3.1, Protune policies conform to given patterns. It is therefore intuitive that ACE policies (meant to be translated into Protune policies) also must conform

[4] http://attempto.ifi.uzh.ch/site/

to predefined patterns. It is indeed the case since, roughly speaking (more on this in [18]), ACE policies must have one of the following formats.

- If $condition_1$ and ... and $condition_n$ then *action*.
- If $condition_1$ and ... and $condition_n$ then *condition*.

where $n \geq 0$, *condition* and $condition_i$ $(0 \leq i \leq n)$ are phrases containing the features listed above (e.g., `the requester is a friend` or `the resource is a family-picture`) and *action* is a phrase containing (beside the features listed above) the modal verb `can` (e.g., `the requester can access the resource`). As the reader can see, all policies presented in this section as well as those listed in Table 1 and 2 conform to these patterns.

After having introduced the ACE→Protune mapping in an intuitive way, we conclude this section by providing a more principled approach to the translation by listing the most remarkable mapping rules we defined.

A programmer asked to formalize the sentence *John sends Mary the credential* as a logic program would most likely come up with a rule like $send(john, mary, credential)$. Indeed in many cases translating verbs into predicate(name)s can be considered the most linear approach, which we pursued as well. However this approach only allows to produce predicates with a number of parameters up to three. In order to support predicates with a bigger number of parameters we decided to exploit ACE propositional phrases. For instance, sentence *If ... then John pays the book with the credit-card.* is translated to $'pay\#with'('John', Book, Credit\_card) \leftarrow \ldots$

The statement *John is Mary's friend* can be seen as asserting some information about entity "Mary", namely that the value of her property "friend" is "John". It should be then intuitive exploiting Protune complex terms (recall Section 3.1) to map such ACE sentence to $'Mary'.friend :' John'$.

When translating noun phrases like *a user* it must be decided if it really matters whether we are speaking about a "user" (in which case the noun phrase could be translated as $user(X)$) or not (in which case the noun phrase could be translated as a variable X). According to our experience, policy authors tend to use specific concepts even if they actually mean generic entities. For this reason we followed the second approach, according to which the sentence *If a user owns a file then the user can access the file.* is translated into the Protune rule: $allow(access(User, File)) \leftarrow own(User, File)$. If it is needed to point out that the one owning a file is a user, the sentence can be rewritten e.g., as follows: *If X is a user and X owns a file then X can access the file.* which gives the translation: $allow(access(X, File))$ $\leftarrow user(X), own(X, File)$.

In the following section we will detail how such natural language policies can be used for privacy preservation in a social platform.

5 Policy Enforcement for Preserving Privacy in Social Platforms

Using policies and the policy framework Protune for access control provides the user with a fine-grained specification of his privacy preferences in a declarative manner. So far we explained how natural language eases the specification of access control policies. In this section we explain how enforcing those policies preserves privacy in social data sharing platforms.

ACE Policy	Protune Policy
P1 If a document is protected then if the requester sends a student-id and the student-id's issuer is "uni-hannover" then the requester can access the document.	$allow(access(Requester0, Document0)) \leftarrow$ $protected(Document0),$ $send(Requester0, Student\_id0),$ $Student\_id0.issuer :' uni\_hannover'.$
P2 Every requester can access everything which is an employee-credential.	$allow(access(Requester0, Something0)) \leftarrow$ $'employee\_credential'(Something0).$

Table 1. A lecturer's policy, in CNL and its corresponding Protune translation.

ACE Policy	Protune Policy
P3 If a document is protected then if the requester sends an employee-credential C and the issuer of C is "uni-hannover" then the requester can access everything which is a student-id.	$allow(access(Requester0, Something0)) \leftarrow$ $send(Requester0, Employee\_credential0),$ $Employee\_credential0.issuer :' uni\_hannover',$ $'student\_id'(Something0).$

Table 2. A student's policy, in CNL and its corresponding Protune translation.

Since policies are declarative statements and by using a current state-of-the-art policy engine, we allow advanced control of data sharing. We will now shortly name those features [19]—instantiated as the features of Protune—and describe their benefits for social software:

Negotiations: In traditional access control or authorization scenarios only one party is able to specify policies which the other one has to conform to. Typically only one of the interacting parties is enabled to specify the requirements the other has to fulfill, whereas the other has no other choice but satisfying them (and thereby being authorized) or not (and thereby not being authorized). Therefore, a more expressive approach allows both parties to discuss (i.e., negotiate) in order to reach an agreement. Trust Negotiations [3], for example, allow both parties to incrementally disclose information in order to get or grant access to some data on a social platform. As an example, see the policies in the Tables 1 and 2: if the student requests access to a lecturers document that is protected, she is asked to provide her student id. But the student herself protects her student id by Policy P3 stating that any party that requests the student id has to be an employee of the University of Hannover. Consequently, the lecturers request for a student id leads to the student's request for the lecturer's employee credential. Since this credential is not protected (see Policy P2), it is disclosed, therefore the student discloses the student id and is allowed to access the protected document. Mind that, first, this procedure is automatically performed by the parties' Protune clients and that, second, without such a negotiation, the student would not have got access to the document.

Evaluation and Actions: In order to evaluate a policy, some actions performed by the policy infrastructure might be required. Examples for such actions are sending credentials or queries to external systems, for example in order to decide if the requester is a registered friend on some other social platform. In this case, the Protune policy enforcement infrastructure can automatically query this platform to get the required information. Another common action is the retrieval of environmental properties like the current system time (e.g., if access is allowed only in a specific time frame) or location (e.g., share files only with people in the same meeting room). By doing this, privacy control in social platforms becomes extensible and information from other accessible social platforms can be exploited for security decisions.

Explanations: As stated in Section 3.1, with Protune it is possible to generate explanations [20] out of the policies and the decisions they make. On the one hand, this helps a user to check whether the policies she created are correct and, on the other hand, they inform other users about why a decision was taken (or how the users can change the decision by performing a specific action). For example, whenever access to a certain data or resource is denied on a data sharing platform, an explanation why the request failed can be provided. Given the example from Tables 1 and 2, a requester that does not disclose her student id would get an explanation such as "I cannot find a Credential such that Credential is a student id." Together with the authoring of policies in CNL, such natural language explanations encapsulate the formal policy evaluation completely from the user.

Strong/lightweight evidences: The result of a policy's evaluation may depend on the identity or other properties of a requester such as age, membership in a certain club, etc. Protune provide a means to communicate such properties. Usually, this is done by sending digital certificates called *strong evidences*. Typical strong evidences are credentials signed by trusted entities called *certification authorities*. *Lightweight evidences* [21] are non signed declarations or statements (e.g., license agreements).

Ontologies: In trust negotiations as they are described above, policies have to be exchanged among entities within the social platform in order to transfer information about what is needed to fulfill a policy. For example, the lecturer's Protune client in the example has to transfer the policy protecting the document in order to let the student's client know that a student id is required. Although the basic constructs may be defined in the policy language (e.g., rule structure and semantics), policies may be used in different applications and even define new concepts. The ontology support of Protune helps to provide well-defined semantics for new concepts to be understood by different entities[22, 23].

All those features of our approach meet the requirements identified in Section 2 as follows. Using the Protune language makes our privacy solution adaptive and fine-grained. Protune's ability to add external data sources allows to extend our solution towards data sources that lay outside of the actual social platform. The natural language policy definition and Protune's explanation facility makes our approach talk natural language to the user and vice versa. Last but not least, our solution comprises the common security features. The logic programming nature of the policy language behind makes the privacy enforcement reliable.

6 Implementation

In this section we present a prototype implementing privacy control in a collaborative platform with natural language policies.[5] We realized the presented approach as an extension to the collaborative learning environment LearnWeb2.0 [24]. LearnWeb2.0 is a Web2.0 application which integrates several Web2.0 tools and allows the user to collaboratively develop new competences while staying in a single comfortable environment. Therefore, LearnWeb2.0 supports sharing and rating of arbitrary resources that help developing competences.

The extension we describe in this paper allows users to exactly define which condition a user has to meet in order to access a resource that is shared on the platform. These policies can be defined in controlled natural language policies and are subsequently translated into Protune policies and enforced whenever a user requests access to a resource belonging to a

[5] The prototype is accessible at http://learnweb.L3S.uni-hannover.de.

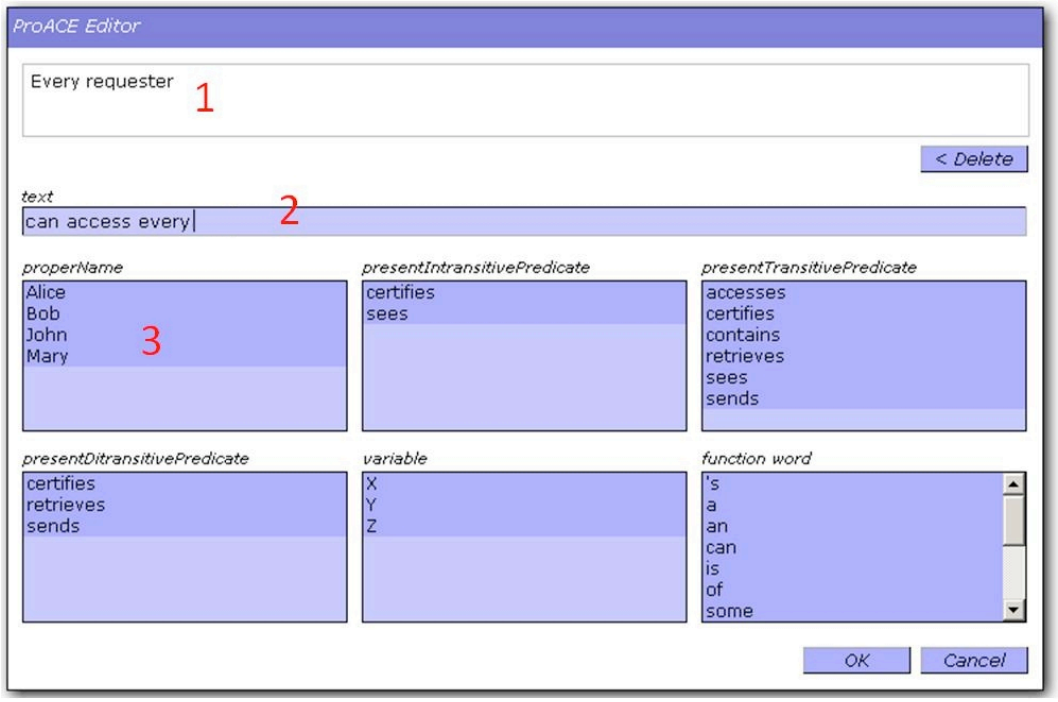

Fig. 1. Creating a policy with the ACE policy editor.

collaborator. Before we detail our extension, we first shortly introduce the features of the LearnWeb2.0 platform.

6.1 LearnWeb2.0

The main goal of LearnWeb2.0 is to help the user to organize, rate, and manage resources that support the development of her desired competences. In order to create one's own specific set of learning resources, there are three ways to add resources to the repository. First, the user search via the LearnWeb2.0 search utility for publicly available content in a collection of Web2.0 tools such as YouTube, Flickr, Ipernity, GroupMe!, and Delicious. The user can then select resources that correspond to the desired competence and add them to her LearnWeb2.0 repository. Second, LearnWeb2.0 offers the user a virtual desktop facility with access to all resources (including private ones) stored on different Web2.0 applications in her own accounts. Third, resources can be directly uploaded from the user's desktop. Therefore LearnWeb2.0 offers a common upload interface which stores the uploaded resources on suitable Web2.0 accounts that are connected to LearnWeb2.0. Given this upload feature, a user can store pictures on Flickr, videos on YouTube, audio tracks on Ipernity and the like from one single platform, namely LearnWeb2.0. Furthermore, resources can be commented, tagged and rated by other LearnWeb2.0 users who are authorized to access them according to the resource owner's policy. Thus a group of users can be established that share learning material and therefore are supported by collaboratively developing a particular competence.

6.2 The ACE Policy Editor

Fig. 1 is a screenshot of the ACE Policy Editor a user can use in order to define Protune policies by means of ACE sentences. This editor is based on the predictive authoring tool[6] developed by the ACE group. The ACE Policy Editor guides the user step by step during the creation or modification of policies and makes sure that only sentences are created which can be translated into Protune policies. Therefore the translation of such sentences into Protune policies will never lead to errors.

The text field marked with (1) is read-only and shows the beginning of an ACE policy. This field is automatically updated whenever the user deletes the last inserted token (by pressing the button "Delete") or add a new token which has been accepted by the editor (cf. below). The text field (2) can be used for entering the next words of the sentence. Whenever the "OK" button is pressed, such words are moved to the text field (1) up to the point where a word is not a correct continuation of the already entered sentence. From this point on, the remaining of the text is kept in (2). Clicking on the entries of the lists (3) is an alternative way to construct a sentence. Each and all words which can be entered at the current position of the sentence are shown according to their grammatical category. In the depicted case, only proper names, intransitive predicates, and so on are allowed. The list for common nouns, for example, is not shown because common nouns are not allowed to appear behind the common noun *requester* in the depicted sentence.

6.3 Extending LearnWeb2.0 with Natural Language Privacy Policies

We extended LearnWeb2.0 by a policy management dialog (see Figure 2). For each policy the title and the text in natural language is returned and displayed in the table. The user has a possibility to edit each of them, or add a new one by supplying a policy title to the creation form and clicking on the submit button. For creating and editing policies LearnWeb2.0 calls the ACE policy editor described in the previous section. The created policies are first translated to Protune policies and then submitted to a Protune server responsible for the policy evaluation.

Policy enforcement is integrated by quering the Protune server whenever a user requests access to another user's resource. For example, in the search function of LearnWeb2.0, only those resources are accessible for the user where the policy evaluation succeeded. Currently, the result of a policy evaluation from Protune server we are using is restricted to boolean decisions. That is, either a policy request fails or does not. Therefore, currently, trust negotiations are not possible since they require more advanced evaluation results describing the required steps that make the negotiation succeed.

7 Conclusions and Further Work

In this paper we describe a flexible solution for access control in social platforms based on controlled natural language policies. This approach allows users to exactly define with whom they want to share their data whereas current solutions require the user to fill out predefined forms with only a subset of the possible privacy conditions. Moreover, we allow the user to

[6] http://attempto.ifi.uzh.ch/aceeditor/

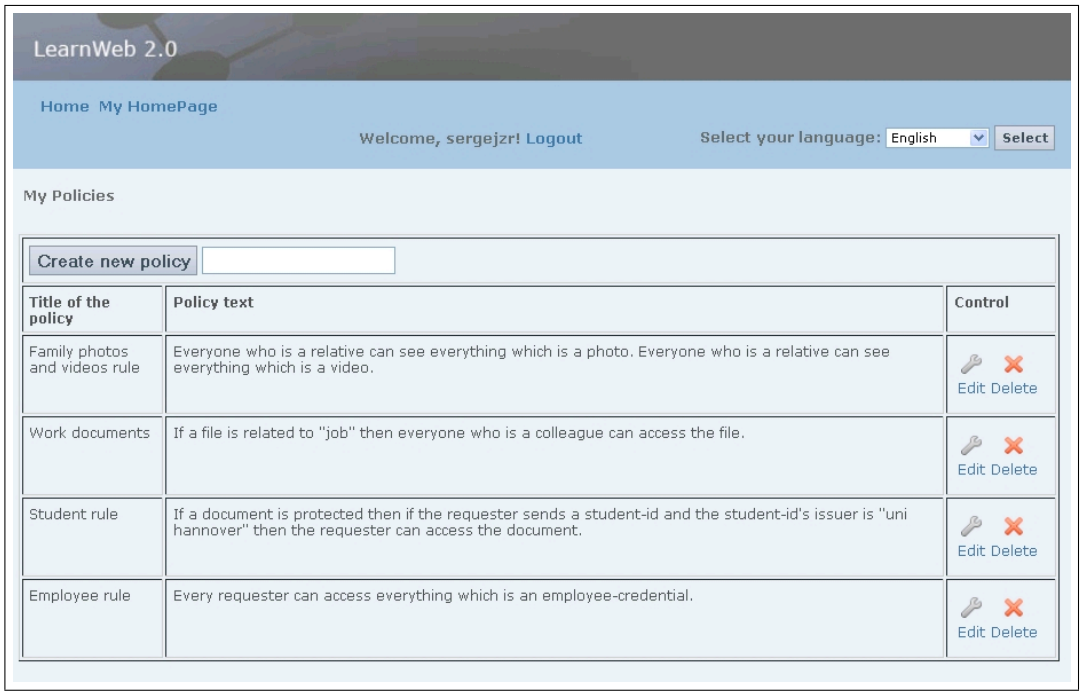

Fig. 2. The LearnWeb privacy settings featuring natural language policies.

define such policies in controlled natural language. We implemented this approach as an extension to the collaborative environment LearnWeb2.0 based on the policy framework Protune and the controlled natural language ACE.

As future work, we plan to extend our implementation to serve for all feature provided by Protune such as explanations and negotiations. The translation of natural language policies to Protune policies still requires careful extensions toward meta-policies. Finally, extensive user studies are on our agenda.

References

1. Passant, A., Kärger, P., Hausenblas, M., Olmedilla, D., Polleres, A., Decker, S.: Enabling trust and privacy on the social web. In: W3C Workshop on the Future of Social Networking, Barcelona, Spain (January 2009)
2. Gross, R., Acquisti, A., Heinz, III, H.J.: Information revelation and privacy in online social networks. In: WPES '05: Proceedings of the 2005 ACM workshop on Privacy in the electronic society, New York, NY, USA, ACM (2005) 71–80
3. Winslett, M.: An introduction to trust negotiation. In: iTrust. (2003) 275–283
4. Nakashima, E.: Feeling betrayed: Facebook users force site to honor their privacy. The Washington Post (November 30, 2007)
5. AP with Asher Moses: Virgin sued for using teen's photo. The Sydney Morning Herold (September 21, 2007)

6. Chew, M., Balfanz, D., Laurie, B.: (under)mining privacy in social networks. In: Web 2.0 Security and Privacy (in conjunction with IEEE Symp.on Security and Privacy). (2008)
7. Bonatti, P.A., Duma, C., Fuchs, N., Nejdl, W., Olmedilla, D., Peer, J., Shahmehri, N.: Semantic web policies - a discussion of requirements and research issues. In: ESWC. Volume 4011 of Lecture Notes in Computer Science., Budva, Montenegro, Springer (2006)
8. Tonti, G., Bradshaw, J.M., Jeffers, R., Montanari, R., Suri, N., Uszok, A.: Semantic web languages for policy representation and reasoning: A comparison of KAoS, Rei, and Ponder. In: International Semantic Web Conference. (2003) 419–437
9. 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. POLICY 2003 (2003) 93
10. Kagal, L., Finin, T.W., Joshi, A.: A policy based approach to security for the semantic web. In: ISWC 2003, Springer (2003)
11. Gavriloaie, R., Nejdl, W., Olmedilla, D., Seamons, K.E., Winslett, M.: No registration needed: How to use declarative policies and negotiation to access sensitive resources on the semantic web. In: ESWS 2004, Heraklion, Crete, Greece, Springer
12. Bonatti, P.A., Olmedilla, D.: Driving and monitoring provisional trust negotiation with metapolicies. In: 6th IEEE Policies for Distributed Systems and Networks (POLICY 2005), Stockholm, Sweden, IEEE Computer Society (June 2005) 14–23
13. Becker, M.Y., Sewell, P.: Cassandra: Distributed access control policies with tunable expressiveness. In: POLICY'04: Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks. (2004) 159–168
14. Coi, J.L.D., Olmedilla, D.: A review of trust management, security and privacy policy languages. In: Int. Conf. on Security and Cryptography (SECRYPT), INSTICC Press (July 2008)
15. Fuchs, N.E., Kaljurand, K., Kuhn, T.: Attempto Controlled English for Knowledge Representation. In Baroglio, C., Bonatti, P.A., Małuszyński, J., Marchiori, M., Polleres, A., Schaffert, S., eds.: Reasoning Web, Fourth International Summer School 2008. Number 5224 in Lecture Notes in Computer Science, Springer (2008) 104–124
16. Grosof, B.N.: Courteous logic programs: Prioritized conflict handling for rules. Technical report, IBM T.J. Watson Research Center (December 1997)
17. Baral, C.: Knowledge Representation, Reasoning and Declarative Problem Solving. Cambridge University Press (2003)
18. De Coi, J.L.: Notes for a possible ACE → Protune mapping. Technical report, Forschungszentrum L3S, Appelstr. 9a, 30167 Hannover (D) (July 2008)
19. Coi, J.L.D., Kärger, P., Koesling, A.W., Olmedilla, D.: Control your elearning environment: Exploiting policies in an open infrastructure for lifelong learning. IEEE Transactions on Learning Technologies **1**(1) (2008)
20. Bonatti, P.A., Olmedilla, D., Peer, J.: Advanced policy explanations on the web. In: 17th European Conf. on Artificial Intelligence (ECAI), Italy, IOS Press (2006)
21. Bonatti, P., Samarati, P.: Regulating service access and information release on the web. In: CCS, ACM Press (2000) 134–143
22. Leithead, T., Nejdl, W., Olmedilla, D., Seamons, K.E., Winslett, M., Yu, T., Zhang, C.C.: How to exploit ontologies for trust negotiation. In: ISWC Workshop on Trust, Security, and Reputation on the Semantic Web. CEUR Workshop Proceedings (2004)
23. Nejdl, W., Olmedilla, D., Winslett, M., Zhang, C.C.: Ontology-based policy specification and management. In: ESWC, Heraklion, Greece, Springer (2005)
24. Marenzi, I., Demidova, E., Nejdl, W., Zerr, S.: Social software for lifelong competence development: Challenges and infrastructure. International Journal of Emerging Technologies in Learning (iJET) (2008)

Privacy Protection on the Social and Semantic Web

Trust and Privacy on the Social and Semantic Web (SPOT 2009)

FOAF+SSL: RESTful Authentication for the Social Web*

Henry Story[1], Bruno Harbulot[2], Ian Jacobi[3], and Mike Jones[2]

[1] Sun Microsystems, http://blogs.sun.com/bblfish
[2] The University of Manchester, UK, Bruno.Harbulot@manchester.ac.uk
[3] MIT

Abstract. We describe a simple protocol for RESTful authentication, using widely deployed technologies such as HTTP, SSL/TLS and Semantic Web vocabularies. This protocol can be used for one-click sign-on to web sites using existing browsers — requiring the user to enter neither an identifier nor a password. Upon this, distributed, open yet secure social networks and applications can be built. After summarizing each of these technologies and how they come together in FOAF+SSL,[4] we describe declaratively the reasoning of a server in its authentication decision. Finally, we compare this protocol to others in the same space.

1 Introduction

Many services that require authentication rely on centralized systems. The identity of the user is constrained to that administrative domain, forcing her to have a different account and identifier for each organization she interacts with. This inability to relate identities easily across domains also makes the creation of links between people in distinct organizations difficult.

Every time a person needs authenticated access to a new organization, a new registration needs to be made; this is a burden for both the user and the organization. The process of registration is either (a) minimal — for example, e-mail address confirmation —, or (b) more formal — for example, in a workplace, where an administrator has to create an account after making verifications out-of-band. Process (a) is lightweight, but will often provide insufficient information, whereas process (b) may initially be able to give more information about a user, at the expense of a costly verification phase during the registration.

Attempts to decentralize this process have been made. Shibboleth,[5] for example, aims at sharing accounts across administrative boundaries; however, it relies on a rigid federation process between organizations. OpenID, enables authenticating a user against a URI, but does not build a social web on that.

[4] Up-to-date information on developments in this protocol are available at http://esw.w3.org/topic/foaf+ssl.

[5] http://shibboleth.internet2.edu/

Neither Shibboleth nor OpenID fully comply with web architecture principles (see Section 2.1 on REST), which in part explains their limitations.

This paper describes a novel approach that relies on combining the use of SSL client certificates and Semantic-Web-based FOAF networks. The result is a secure, open and distributed authentication mechanism, which is able to satisfy simple requirements — such as authenticating a user by URI, like OpenID, but without the user needing to remember this URI — as well as more complex requirements, where the authorization to a service depends on distributed properties of the user, such as his position in a social network. This proposal is built on a RESTful architecture, the same that underpins the largest and most successful network of distributed linked information, the Web.

Section 2 introduces the background technologies of the Semantic Web and FOAF, as well as cryptography and client-certificate authentication. Section 3 presents the FOAF+SSL protocol. Section 4 compares this approach to others.

2 Background

2.1 The RESTful Web Architecture

Representational State Transfer (REST) [1, Chap. 5] is an architectural style for building large-scale distributed information networks, the most famous of these being the World Wide Web [2]. To build such a network requires that each of the parts be able to grow independently of any of the others, with very little central coordination, and that each of the resources thus created be able to refer easily to any of the others. The logical building blocks for this are the following:

1. The specification of universal names, also known as Universal Resource Identifiers (URIs) — such as the familiar Universal Resource Locators (URLs).
2. The mapping of URIs to Resources, also known as the reference relation.
3. Canonical methods for manipulating the resources mapped by each URI, via *representations* of the resource. Such a protocol specifies a canonical dereferencing mechanism, enabling a holder of a URI to find and manipulate the resource referred to by that URI. `http://...` URLs use the HTTP protocol as their dereferencing mechanism, for example. By sending a GET request to the object at a given HTTP URL, a *representation* of the resource is returned. A resource can be created, changed, and deleted using the POST, PUT, and DELETE methods respectively.

REST specifies the architectural style required to build such a protocol with the aim of maximum networkability; that is, any representation should be able to link to any resource from anywhere, using the URI alone to do so.

2.2 The Semantic Web

Whereas URLs in the initial web of hyperlinked documents referred only to documents, the Semantic Web specifies how to extend this to enable a web of resources. In the Semantic Web, it becomes possible for URLs to refer to anything, be it:

1. physical things — for example, `<https://romeo.example/#i>` may refer to a human named Romeo;
2. relations between two individuals — for example, the relation of knowing someone which `<http://xmlns.com/foaf/0.1/knows>` refers to; or
3. classes — for example, people may be described as being instances of `<http://xmlns.com/foaf/0.1/Person>`, the set of persons.

The meaning of these URLs can be found by dereferencing them using their canonical protocol. Thus, doing an HTTP GET on `<http://xmlns.com/foaf/0.1/knows>` should return a representation describing it. Since HTTP is built to allow content negotiation, well configured web servers will return the representation best fitting the client's needs. Entering the above URL in a web browser will return a human readable HTML page describing the 'knows' relation. A Semantic Web agent could ask for the standard machine-friendly RDF representation.

A Semantic Web document is a serialization of a graph of directed relations between objects. Each relation exists as a triple of `<subject> <relation> <object>`, where each of `subject`, `relation` and `object` can be chosen among any of the URIs or string literals. Since it is tedious to read and write such URLs, this article uses the N3[6] `@prefix` notation. The following prefixes will be used throughout this article:

```
@prefix log: <http://www.w3.org/2000/10/swap/log#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix cert: <http://www.w3.org/ns/auth/cert#> .
@prefix rsa: <http://www.w3.org/ns/auth/rsa#>
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix romeo: <https://romeo.example/#> .  #see RFC2606 on .example domains
@prefix jult: <https://juliet.example/#> .
@prefix : <> . # special vocabulary defined for this paper
```

Thus, to say "Romeo is a person", one can write: `romeo:i a foaf:Person.`. Since each of the resources in that sentence is named by a URL, one can GET further information about each by dereferencing it, and if that representation itself contains relations do the same recursively. This is known as Linked Data[7].

Each representation returned by a resource can be interpreted as a graph of relations, which can be isolated in N3 by placing them within curly brackets { }.[8] The relation that maps a resource to the graph described by the document retrieved using the canonical dereferencing method of its URI is defined as the `:semantics` relation. Thus, after dereferencing `romeo:i`, one may state the following, without asserting the actual statements within the brackets as true:

(P1)

```
romeo:i :semantics { romeo:i a foaf:Person;
                        :hasPrivateKeyFor pubKey;
                        foaf:name "Romeo";
                        foaf:knows  jult:me . }
```

[6] Current N3 tutorial at: http://www.w3.org/2000/10/swap/doc/Overview.html.
[7] see Tim Berners-Lee's note http://www.w3.org/DesignIssues/LinkedData.html
[8] Unlike the Named Graph brackets in SPARQL; N3 supports anonymous graphs.

These graphs can also be used to formulate rules, as when we define the above `:semantics` relation in terms of the established `log:semantics` property, which relates a document to its graph, and `:representation` relating a resource to one of its representations (the `?` prefixed variables are universally quantified):

(D1)
```
{ ?resource :representation ?doc . ?doc log:semantics ?graph . }
      => { ?resource :semantics ?graph . }
```

The `log:` namespace[9] tends to make significant use of enclosed graphs, or "formulas". In particular, the `log:includes` property links a subject graph to an object graph by asserting that the latter is a subset of the former graph; the `log:implies` property, also written as `=>`, can serve as the basis for reasoning based on first-order logic (with the introduction of appropriate variables).

Even though the Semantic Web is built in order to make merging of information easy, it is not a requirement to do so. We will be using this notation to help illustrate clearly when merging graphs is reasonable.

2.3 FOAF, reputation networks and the Web of Trust

FOAF,[10] short for Friend-of-a-Friend, is an RDF vocabulary used to describe people, agents, groups and their relations. When used on the Semantic Web, this allows each person to describe himself and his network of friends.

By giving oneself a URI — *aka.* a Web ID —, one can describe one's personal social network by linking oneself to acquaintances by reference. Someone who has been given the `romeo:i` URL by Romeo himself, and then fetched its `:semantics` (ending up with the statements in P1) has good reason to trust that the information there is correct and, thus, to merge it (in a defeasible manner) with his own belief store. This graph itself will contain further URIs, such as `jult:me`, whose `:semantics` the agent can also GET. Similarly, the user can then add `romeo:i` to his FOAF file, to publish `:me foaf:knows romeo:i`.

Thus, a peer-to-peer information network can be built, where each person specializes in keeping up-to-date the information they feel responsible for, linking to the best sources for objects they do not wish to maintain. In return, as the quality of one's information and links increases, others feel more confident linking to it, reducing their own work and responsibilities.

As the network grows, the value of the network grows exponentially, as predicted by Metcalf's Law [3], creating a virtuous circle. Current social networking sites, such as Facebook and LinkedIn, to name a few among many, have shown how this can work in less distributed settings, taking advantage of the same law.

The plain `foaf:knows` relation may be enhanced with trust descriptions so as to create a *reputation network* [4], and, in the case of FOAF+SSL, this trust can be backed by the use of cryptographic keys and signatures, so as to form a secure Web of Trust (as described in the next sections).

[9] The `log:` namespace is described at http://www.w3.org/DesignIssues/N3Logic.
[10] Defined at http://xmlns.com/foaf/0.1/.

2.4 Public key cryptography

Public key cryptography allows two peers to communicate securely without requiring them to share a secret, through the use of unique pairs of keys. One key, called the *public key*, may be disseminated widely, and the other, the *private key*, is to be kept only by its owner. This is in contrast to symmetric cryptography, where both participants must share the knowledge of the same *secret* key for both encryption and decryption.

Public key cryptography relies on the conjecture that it is infeasible to obtain any private key that corresponds to a given public key through brute force because this operation is too computationally expensive. It also assumes that no two distinct individuals will generate the same key-pair randomly. We can define in D2 an inverse functional property `:hasPrivateKeyFor`.

(D2)
```
:hasPrivateKeyFor a owl:InverseFunctionalProperty;
        rdfs:domain foaf:Agent;
        rdfs:range cert:PublicKey .
```

Thanks to the dual-nature of the public and private key pair, two distinct actions are made possible:

1. *Encryption* is the obfuscation of a plain text message, generating a scrambled message using the public key of a key pair, so that it may only be decrypted using the corresponding private key.
2. *Signing* is the process of associating a digital signature with a message; this signature is generated using a private key. The authenticity and integrity of the message can then be verified using the corresponding public key.

A *public key certificate* is the signed combination of a public key and some information related to this key. Such a certificate may be self-signed (using the private key that matches the public key it contains) or signed by a third party. The party that signs a certificate (self-signed or not) endorses its contents. Trusting the party that signed a certificate can be a reason for believing its contents.

Two different architectures have been developed to make use of third party signing of public key certificates: the hierarchical Public Key Infrastructure (Section 2.5) and the cryptographic Web of Trust (Section 2.6). In both architectures, an application or hosting environment is initially configured with a trusted set of certificates known as *trust anchors*. When presented with an unknown certificate, an application verifies its authenticity by attempting to build a certification path — or chain — between the certificate and one of the trust anchors. A certificate becomes trusted if and only if it has been signed using a certificate which is already trusted. If necessary, this operation may be repeated to build a path through intermediate certificates, through which the trust relation is transitive.

The hierarchical Public Key Infrastructure model and the cryptographic Web of Trust model mainly differ in the way in which certificates are distributed and intermediates are trusted. In both cases, the initial establishment of trust (i.e. the selection of trust anchors) requires an initial import of certificates which is out of band, but this process is much less onerous than obtaining all public key certificates for all entities likely to take part in secure communications.

2.5 PKI and hierarchical model of trust

The *Internet X.509 Public Key Infrastructure* [5] (PKI) is a hierarchical model for distributing and trusting certificates. In this model, certificates are signed by a *certification authority* (CA). X.509 certificates incorporate a *Subject Distinguished Name* (Subject DN), which identifies the subject of the certificate, and an *Issuer Distinguished Name* (Issuer DN), which identifies the issuer of the certificate — the entity that signs the certificate . An X.509 certificate may only have one Issuer DN, which must be the Subject DN of the certificate that has been used to issue it. This structure builds a hierarchical tree from the root CA certificate, via optional intermediate CA certificates, to the end-entity (i.e. client or server) certificates.

Most web-browsers and operating systems provide a default list of CA certificates which they make their users trust implicitly. This list can usually be changed by the user, a feature often used by institutional PKIs.

2.6 Cryptographic Web of Trust

The cryptographic *Web of Trust* (WoT) is a form of public key infrastructure (although rarely called PKI) where each participant may assert trust in any other participant, without a specific hierarchy.

The Web of Trust model is used by PGP; what is often referred to as a *PGP public key* is, in fact, a form of a public key certificate, since it also contains additional information (such as an e-mail address) and is signed so as to assert its authenticity. Such a certificate is self-signed, but may also contain additional signatures — those by whom the association between the key and this additional information is trusted.

In PGP, the trust anchors are the user's own certificate and the certificates the user trusts, some of which may be from trusted introducers (that is, people through whom trust is transitive). The number a signatures a certificate has reflects the connectivity to other parties. The more signatures a certificate has, the more likely it is that a third party will be able to find a certification path to that certificate via trusted introducers.

2.7 SSL authentication

The most widely deployed protocol for securing communications between a user-agent and a web server is Transport Layer Security (TLS) [6], itself a successor to the Secure Socket Layer 3.0 (SSLv3) specification;[11] its use in HTTP applications is denoted by the `https` prefix in URLs.

During the SSL handshake, at the beginning of the SSL connection, the client obtains an X.509 certificate from the server. At this point, the client relies on its trust anchors to verify it. If this certificate is trusted and verified, the handshake proceeds. Once the handshake has finished, the communication (on

[11] Unless explicitly noted, this article uses SSL to encompass TLS 1.x and SSL 3.0.

top of SSL) can proceed in a secure manner; the only other party capable of reading the communication must have the private key corresponding to this server certificate.

There exists a variant of the handshake procedure in which the client is requested or required to present a certificate to the server, enabling the server to authenticate the client using the same verification method as above.

The remainder of this section describes, from a Semantic Web point of view, how trust in a certificate is evaluated. This forms the basis for comparison of how FOAF+SSL differs from this, in Section 3.2. We describes the reasoning of a server, `S`, for authenticating a client, `:client`, making a request. Server `S` has a set of trusted CAs. `S` would state that `issuerDN` was a trusted CA with:

(P2)
```
issuerDN a :TrustedCA;
         :hasPrivateKeyFor CAKey .
```

The `CAKey` is a `cert:PublicKey` that is usually identified by a number of inverse functional properties, which form an OWL2 key.[12] For the sake of brevity, these relations are not shown here. Suffice it to say that CA Keys can be uniquely identified by them.

`S` requests that `:client` presents a certificate signed by any one of a number of CAs it knows about. `S` receives `_:certDoc` with semantics such as the following (the subject is also identified via a DN):

(P3)
```
_:certDoc :semantics _:certSemantics .
_:certSemantics = { <> dc:created issuerDN;
                       foaf:primaryTopic subjectDN .
                    subjectDN :hasPrivateKeyFor pubKey .
                    issuerDN :hasPrivateKeyFor CAKey .  }
```

So far, the SSL handshake ensured `S` that the client has the private key:

(P4)
```
:client :hasPrivateKeyFor pubKey .
```

The client asserts the contents of the certificate (not shown) and that it is signed by `issuer`:

(P5)
```
:client :claims  { _:certDoc :signature _:certSig;
                   _:certSig :signedWith CAKey;
                             :sigString "XYZ SIG" . }
```

`S` can assert, after verification, that `_:certDoc` has been signed using the private key corresponding to `CAKey`:

(P6)
```
_:certDoc :signature [ :signedWith CAKey ] .
```

Proving that a document is signed by P, is to assert P claims its contents:

(D3)
```
{  ?P :hasPrivateKeyFor ?key .
   ?doc :signature [ :signedWith ?key ]
} => {  ?P :claims [ is :semantics of ?doc ] } .
```

[12] http://www.w3.org/TR/owl2-syntax/#Keys

Then, from the signature verification P6, the certificate contents P3 and the definition D3, `S` can assert:

(P7) `issuer :claims _:certSemantics .`

To trust someone is to trust what they claim. `S` trusts `TrustedCA`s, thus:

(D4) `{ ?ca :claims ?s . ?ca a :TrustedCA } => { ?s a log:Truth } .`

From P2, P7 and D4, `S` can conclude:

(P8) `subjectDN :hasPrivateKeyFor pubKey .`

From P4 gained by the SSL handshake, the above P8 and the definition D2 of `:hasPrivateKeyGot` as a `owl:inverseFunctionalProperty`, we can deduce:

(P9) `:client owl:sameAs subjectDN .`

At this point, the server `S` has authenticated the `:client` as this Distinguished Name (DN). Considering that the `:client`'s request is to access resource `R`, the server can then find out if this DN is authorized access to `R`.

The problem with DNs is that, although they can be made to form a URI, the dereferencing mechanism for DNs is not global in the way `http` URLs are. Therefore, if access to `R` is granted in some rule based way, where more information about `R` needs to be discovered for a decision to be made, then the DN cannot provide a global solution. For very much the same reasons, data in LDAP servers cannot have fields that point resources in any other LDAP server. As a result, current uses of client certificates limit the usage of each to a few domains.

The ability to link globally is an essential piece required for building a global social network. The next sections shows how FOAF+SSL solves this problem.

3 The FOAF+SSL protocol

This section describes the FOAF+SSL protocol. The FOAF+SSL protocol uses SSL but uses a different trust model than PKI to verify certificates.[13]

When protecting a service, it is important to differentiate authentication from authorization. Authentication is the action of verifying the identity of the remote user. Authorization consists of allowing or denying access to or operations on a given resource, based on the identity obtained during authentication.

FOAF+SSL enables a server to authenticate a client given a simple URL. This URL can then be used directly for authorization, or to explore more information in the web of linked data, in order to decide if the referent of the URL satisfies the constraints required for accessing the resource.

[13] Although the examples we use are based on the Web, FOAF+SSL could in principle also be used for authentication to other SSL-enabled services, such as IMAPS.

3.1 Protocol sequence

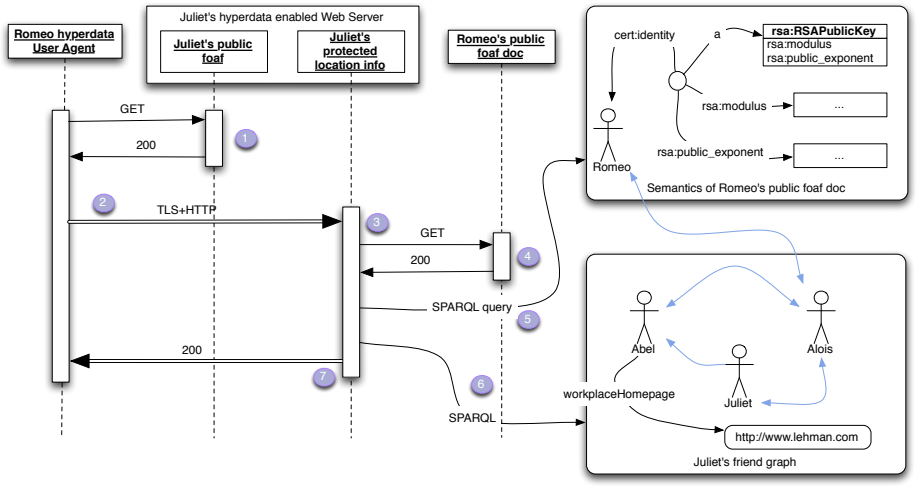

Fig. 1. The FOAF+SSL sequence diagram.

The FOAF+SSL authentication protocol consists of the following steps, as illustrated in Figure 1:

1. A client fetches a public HTTP resource which points to a protected resource, for example `<https://juliet.example/location>`.
2. The client, `romeo:i`, dereferences this URL.
3. During the SSL handshake, the server requests a client certificate. Because FOAF+SSL does not rely on CAs, it can ask for *any* certificate. The client sends Romeo's certificate (which may be self-signed) containing its public key (see "`Subject Public Key Info`" in Listing 1.1) and a *Subject Alternative Name* URI (see "`X509v3 extensions`" in Listing 1.1). Because the SSL handshake has been successful, Juliet's server knows that Romeo's client has the private key corresponding to the public key of the certificate.

Listing 1.1. Excerpt of a text representation of a FOAF+SSL certificate.

```
    Subject Public Key Info:
           Public Key Algorithm: rsaEncryption
           RSA Public Key: (1024 bit)
               Modulus (1024 bit):
                   00:b6:bd:6c:e1:a5:ef:51:aa:a6:97:52:c6:af:2e:
                   71:94:8a:b6:da:9e:5a:5f:08:6d:ba:75:48:d8:b8:
                   [...]
               Exponent: 65537 (0x10001)
    X509v3 extensions:
          X509v3 Subject Alternative Name:
                           URI:https://romeo.example/#i
```

4. Juliet's server dereferences the Subject Alternative Name URI found in the certificate and ends up with a superset of D1.
5. The document's *log:semantics* is queried for information regarding the public key contained in the X.509 certificate. This can be done in part with a SPARQL query as shown in Listing 1.2. If the public key of the certificate matches the one published in the FOAF file, `romeo:i` is authenticated.

Listing 1.2. SPARQL query to obtain the public key information.

```
SELECT ?modulus ?exp WHERE {
   ?key cert:identity <https://romeo.example/#i>;
        a rsa:RSAPublicKey;
        rsa:modulus [ cert:hex ?modulus; ];
        rsa:public_exponent [ cert:decimal ?exp ] .   }
```

6. Once this authentication step is complete, the position of `romoe:i` in Juliet's social graph can be determined. Juliet's server can get this information by crawling the web starting from her FOAF file, or by other means.
7. Authentication has been done; authorization can now take place.

3.2 Authentication Logic

This section draws a parallel with Section 2.7, again, following the reasoning of the web server `S` trying to authenticate a `:client`.

At the end of stage 3 in the FOAF+SSL sequence diagram, `S` has received the client certificate securely. Being self-signed (or signed by an unknown party), its semantics are somewhat different. `S` is really only interested in the URI identifiers referring to the subject — abandoning thus the limitations of DNs. In addition, since it is asserted by the client, `S` knows that:[14]

(P10)
```
:client :claims { <>  dc:created romeo:i;
                      foaf:primaryTopic romeo:i.
                  romeo:i :hasPrivateKeyFor pubKey .  }
```

`S` may know of `romeo:i` only what it gathered from the SSL handshake:

(P11)
```
:client :hasPrivateKeyFor pubKey .
```

When someone makes a claim, they have to agree with the logical consequences of their claim, including those arising from new facts. Thus, someone who makes a claim `:mustAgree` with the conclusions of the union of what we know securely, what they believe, and established reasoning rules:

(D5)
```
{  ?client :claims ?clientGraph .
 ( ?clientGrph   secureFactGraph   owlReasoningRules )
    log:conjunction [ log:conclusion ?C ] }
             => { ?client :mustAgree ?C  }
```

[14] The signer being the author, following the reasoning from P10, P6, D3 would also end up with this result — for self signed certificates only.

Hence, from P10, P11, and D2, `S` may conclude that `:client` would have to agree that it is `romeo:i`. This should not be a surprise, as that is indeed what one assumes someone who sends such a certificate intends.

(P12) `:client :mustAgree [ log:includes { romeo:i = :client } ] .`

Since `:client` asserts it is `romeo:i`, it accepts to be inspected via `romeo:i`. Since `romeo:i` is a URL, `S` can dereference it and thus discover P1. Then, by P1, P11, D2, and D5:

(P13) `romeo:i :mustAgree [ log:includes { romeo:i = :client } ] .`

In other words, both `romeo:i` and `:client` must agree, given what `S` knows, that `romeo:i owl:sameAs :client`. In particular `romeo:i` cannot repudiate this assertion since `romeo:i` itself provided P1 authoritatively. It follows that if `S` is authorized to serve `R` to `romeo:i`, `S` can serve `R` to `:client`.

3.3 Following links

In the previous section, we showed how a server can authenticate a client who claims a Web ID. Authorization policies — how to decide which group of agents may access which resource — will be particular to each application. It could be done by giving a list of authorized Web IDs. It could be done by trusting statements returned by a selected group of Web IDs, for example, when allowing all friends of a given friend access to a resource, as specified by the representation returned by their Web ID. This would make the initial list of trusted Web IDs similar to the trust anchors in PKI and WoT. A lot still remains to be explored.

A topic for further research is to define the various ways in which trust can be transmitted. Let us look at one simple example here. Imagine a user connects to a service and is authenticated as `joe:i`, using the FOAF+SSL method described above. Imagine `joe:i` returns a representation claiming `joe:i owl:sameAs romeo:i`. Since anyone could make that statement, that, in itself, should not be the basis for belief for services that care about security. If, on the other hand, `romeo:i :semantics [ log:includes { romeo:i owl:sameAs joe:i . } ]`, then this could be used as confirmation of the claim, and from there on both IDs could be used interchangeably by `S`.

4 Related work

Unlike the OpenPGP extension to TLS [7], which also aims to rely on a Web-of-Trust by using PGP certificates instead of the usual X.509 certificates, FOAF+SSL makes only slight changes in the way X.509 certificates are used; it does not require changes in the actual SSL stack. With the OpenPGP TLS extension, the problem of key distribution still remains. Public PGP keys can be stored on public key servers, but there is no global dereferencing mechanism for finding a key, as there is in the FOAF+SSL protocol.

OpenID also shares considerable similarities with FOAF+SSL, due in part to OpenID's reliance on URLs as identifiers, just as FOAF+SSL relies on dereferenceable URIs bearing FOAF data. However, OpenID fails to make much use of the information at the OpenID resource, using it only to find the authorization service. As a result, OpenID requires a much higher number of connections to establish identity — 6 as opposed to 2 — and parts ways with RESTful design in the attribute exchange protocol, loosing thereby the advantages of a networked architecture.

5 Conclusions

FOAF+SSL provides a secure and flexible global authentication system. With public key cryptography at its core, it gains all the security advantages of this technology. Because FOAF+SSL is RESTful and integrates well with the Semantic Web, it can discover more information about an entity by walking the Linked Data cloud. Compared with PKI, FOAF+SSL removes the need for hierarchical authorities to assert identity, making it much more flexible. Thus, this mechanism adapts itself well to the formation and expansion of virtual organisations and distributed social networks.

Acknowledgements

Ian Jacobi acknowledges funding for this project from NSF Cybertrust award #0524481 and IARPA award #FA8750-07-2-0031. Bruno Harbulot acknowledges EPSRC funding under grant EP/E001947/1 (http://www.nanocmos.ac.uk/).

References

1. Fielding, R.T.: Architectural Styles and the Design of Network-based Software Architectures. PhD thesis, University of California, Irvine (2000) http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm.
2. Jacobs, I., Walsh, N., eds.: Architecture of the World Wide Web, Volume One. (December 2004) http://www.w3.org/TR/2004/REC-webarch-20041215/.
3. Hendler, J., Golbeck, J.: Metcalfe's law, Web 2.0, and the Semantic Web. Web Semant. **6**(1) (2008) 14–20 http://www.cs.umd.edu/~golbeck/downloads/Web20-SW-JWS-webVersion.pdf.
4. Golbeck, J., Parsia, B., Hendler, J.A.: Trust networks on the semantic web. In: WWW (Posters). (2003) http://mindswap.org/papers/Trust.pdf.
5. Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., Polk, W.: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280 (Proposed Standard) (May 2008)
6. Dierks, T., Allen, C.: The TLS Protocol Version 1.0. RFC 2246 (Proposed Standard) (January 1999) Obsoleted by RFC 4346, updated by RFC 3546.
7. Mavrogiannopoulos, N.: Using OpenPGP Keys for Transport Layer Security (TLS) Authentication. RFC 5081 (Experimental) (November 2007)

A scalable architecture for peer privacy on the Web

Spyros Kotoulas and Ruud Stegers

Department of Artificial Intelligence, VU University Amsterdam, The Netherlands

Abstract. We consider the privacy implications of exchanging Web data and present the design of a peer-to-peer discovery mechanism with strong privacy guarantees. Its benefits are that it does not rely on a trusted third party, spreads the computational cost among participants and places minimal restrictions on privacy policies. Furthermore, it provides scalability both in terms of system size and level of privacy offered. We outline an implementation that relies on two overlays: a distributed hash table for discovery and an anonymising overlay. Finally, we evaluate the proposed mechanism against a number of privacy threats and analyse its complexity.

1 Introduction

Organizations holding personal information (e.g. government, social networking websites, banks) provide a Web Service where user agents, after authenticating, can retrieve and modify information that concerns them. User agents cannot keep track of the organizations with data that concerns them since the latter is not inserted or maintained by them. *How can these Web Services be located while maintaining the privacy of users and data?* The basic challenges in designing such a system are that the number of such Web Services may be large, thus it is not possible to contact all of them for each query and that user agents and web services should only be able to see data and queries they are authorized to.

Centralized architectures where a single organization maintains an index of all data and enforces access control policies suffers from the following drawbacks: This organization must be completely *trusted* by all participants. Additionally, the **maintenance** of such a gargantuan volume of data would not be an easy, let alone financially viable, task. Finally, the centralized infrastructure should be able to enforce the **access control** and **privacy policies** of the information providers. This inhibits or precludes the use of locally calculated policies and policies that use private criteria. For example, some Web Service wants to authenticate users itself and does not share user passwords with the central index. This is an especially important restriction for the Social Web or any system where we do not have a priori trust agreements [7]. For these reasons, we advocate the use of a peer-to-peer paradigm, where information providers will be able to keep data locally, or at least in the same authority domain and enforce their own privacy and access control policies.

The focus of this work is on *scalable privacy*, with *scalable* referring to both the *size* of the system and the *level* of the privacy guarantees provided. The

threat to the privacy of an individual is related to the ability of an adversary to associate information or other individuals with it. Instead of a one-size-fits-all approach, the level of privacy in our method is tunable. We explore the trade-off between privacy and performance ranging from *high performance*, providing similar performance and slightly better privacy guarantees than current resource discovery systems and search engines to *high privacy*, providing anonymity for the participating parties and non-disclosure of the content, its description or a unique identifier for it. Note that the focus of this work is not on privacy and access control *policies*, but on an *architecture* and *mechanisms* that allow a more open set of such policies.

We propose a Peer-to-Peer framework for private discovery and querying of (Semantic Web) data and develop a method to locate data providers without revealing their location, identity or the descriptions of the content. To this end we describe a distributed scalable index, implemented on top of a Distributed Hash Table (DHT). Additionally we show a method to obfuscate and approximate the descriptors in the index, so as to prevent disclosure of resource descriptions and combine it with an anonymising network to protect the identities of the parties involved. We test a selection of the possible privacy settings against a set of possible attacks and analyse their performance, paying special attention to the trade-off between the additional privacy mechanisms and the associated computational and network overhead.

In section 2 we describe the privacy threats covered by our work. Related work is described in section 3. In section 4 we introduce our architecture for scalable privacy, including a description of the technologies we employ to achieve this. Finally, we analyse our architecture in relation to a set of privacy attacks and the associations defined in section 2.

2 Privacy threats

We present the threats within a discovery/sharing system in terms of the ability of an adversary to associate aspects of content (e.g. a descriptor or the content itself) to a user. We use the term *provider* for a peer that shares content and *seeker* for a peer that consumes content. Content may be anything (e.g. RDF data or Social networking profile or a medical record) while a *content identifier* is a unique identifier for some content (e.g. a cryptographic hash of the content or a URI). A discovery mechanism facilitates the sharing process by allowing a seeker to either locate content or potential providers based on a set of *descriptors* (e.g. keywords or derivatives). The discovery mechanism usually maintains an *index* of descriptors to providers.

Table 1 presents possible privacy impairing association. We observe that: (a) Any association in the top cells is detrimental for privacy (e.g. a mapping from *content* to *identity*). (b) It is possible to derive more sensitive information from less sensitive information, moving from the bottom of the table to the top (e.g. an adversary that knows the set of URIs that appear in a SPARQL query can get a good estimate of the query itself). (c) Even perceived insignificant associations can lead to significant privacy breaches (e.g. two users that are associated with

What		Who
Content (eg. document) Content description (eg. keywords)		Identity (eg real name)
Query (eg. SPARQL query) Query description (eg. set of URIs)	$\Longleftrightarrow$	Location (eg. IP address)
Content ID (eg. GUID) System (usage of)		Any user in the system

Table 1. Privacy impairing associations. This table shows the different associations that can be made between content and identity ordered by descending significance.

similar content IDs have similar interests, thus user profiling is possible). (d) different association types can be combined. For example, an adversary that knows the IP addresses where the system is accessed from (*system - location* association) and has access to an anonymous query log (*query - any user in the system* association). She can infer a (weak) association between queries and IP addresses. We will refer back to this table in section 5.

3 Background

Discovery/sharing systems are widely used both in closed domains and on the public web. Although encryption schemes exist to ensure security, significant challenges concerning privacy and scale remain. Search engines can scale but have abysmal privacy guarantees. Peer-to-peer systems satisfy the basic principle of lacking a central trusted party, but mechanisms to guarantee privacy are not available yet.

We provide an overview of the privacy implications (for seekers and providers) of current discovery/sharing mechanisms in combination with their scalability properties:

Centralized index An index of all content or content descriptions is kept at a collection of centrally managed servers. Information seekers post queries to the index and receive the content, a content identifier or a provider to ask for the content. This can be a *scalable* solution, assuming adequate financial resources to maintain this infrastructure. Nevertheless, it requires that all parties (both seekers and providers) trust this central index with their privacy, since it has complete control and access to all content and queries. Furthermore, providers should entrust the index to correctly enforce the policies specified with the data (if any).

Broadcast-based unstructured overlays Unstructured overlays are a type of peer-to-peer overlay where peers maintain a set of ad-hoc connections to other peers [4]. Queries are broadcasted to all peers, or a significant subset of peers in the system. They provide good privacy guarantees concerning the information providers, since there is no index that can be used to derive privacy sensitive information from. Additionally, every provider maintains full control of it own data. The privacy of the seekers on the other hand is not protected since their queries can be seen by any peer in the network.

Broadcast-based systems are *not scalable* since queries need to be sent to a large subset of the total peers in the system.

Structured Overlays Structured peer-to-peer overlays impose a global structure on the peer connections [6]. They can be used to implement efficient and scalable indexes spread over a large number of nodes. They are *very scalable* but fare *poorly with provider privacy* since the index is readable by a large number of possibly untrusted nodes, which can be used to derive privacy sensitive information about the provider. Any single compromised index node is a direct threat to privacy of the seekers that uses it, since the queries are exposed..

Confidential indexes Confidential indexes aim at hiding information from the indexed content. The general pattern is to introduce noise to the results in order to avoid strong associations. We describe two recent implementions of confidential indexes that are very relevant to our work.

The confidential index proposed by Bawa et al [1] relies on a public index that divides the providers over a set of privacy groups. Within the group, they collaboratively create a bloom filter with the descriptors of their content using a randomized process so as to avoid exposure of the content of individual providers. These filters, along with the list of all providers in the group are sent to a central server, where an index is created that maps bits in the bloom filter to privacy groups. The privacy of the providers is protected since the privacy groups contain providers that have or do not have the requested content (introduction of false positives).

ZERBER [8] uses a set of largely untrusted servers to maintain an inverted index of descriptors. Access control is enforced on the index. To defend against compromised index nodes, no single index server is given enough information to reconstruct a descriptor by itself. To this end, descriptors are encrypted using k out of n cryptography. This means that every descriptor is split and encrypted into n different parts, each owned by a different index node. Since every index node enforces access control locally, as many as k compromised index nodes are required for an adversary to obtain a complete descriptor. For additional protection, the index maps sets of terms instead of single terms to document descriptors (called a merged posting list mechanism). The decrypted descriptors contain the original index terms (so the irrelevant entries returned as result of the merging can be filtered out) and the location of the document. ZERBER^{+R} [9] additionally provides top-k ranked results.

Our own method falls into this category and is further described in section 4.1. The major improvement over these systems is that our system additionally provides anonymity and can scale to a much larger index, since it is maintained by system participants.

Encrypted indexes Encrypted indexes protect the content by encrypting sensitive parts of the index entries. Since this generally requires complex key management schemes, scalability is a problem.

4 Our Architecture

We are using an index-assisted peer-to-peer model similar to the one in [1]. To have their information indexed, providers *(a')* extract a content descriptor (e.g. keywords, hashes of keywords, ...) and *(b')* store it in the index together with an identifier to contact the provider. To locate and access content, seekers need to take the following steps: *(a)* they create a set of content descriptors based on their query (e.g. a set of keywords, hashes of keywords, ...), *(b)* they send this set to the index which in turn *(c)* returns a number of providers which are (possibly) able to provide the information. *(d)* The seeker selects a number of these providers based on local preferences and *(e)* negotiates directly with the provider to retrieve the requested information. Our method places no restriction on the privacy and access control policies for step (e), except that providers need to be able to enforce them themselves locally. The focus of this work is on defining a scalable architecture for this model and develop *mechanisms* to guarantee privacy and anonymity.

4.1 Scalable architecture

We will outline a distributed and scalable implementation. The meaning of the word scalable is two-fold: First, the performance of the system does not deteriorate severely as the number of participants increases. Second, different levels of privacy are supported. The infrastructure that will be presented functions through the synergy of two peer-to-peer overlays: the *Indexing overlay* and the *Anonymising Overlay*. The former provides a global scalable index on top of a DHT. The latter provides a distributed and scalable mechanism to hide the identity of peers, whenever this is required.

Index-assisted query routing The *indexing overlay* is responsible for providing mappings from the set of descriptors in a query to a set providers that possibly have content that matches these descriptors. It should be able to store a very large number of descriptors, thus a scalable implementation is required.

Distributed Hash Tables(DHTs) are a type of structured peer-to-peer overlays that impose a global structure on the peer connections [6]. Typically, each item stored in the DHT is associated with a hash ID chosen from a large key space. This space is partitioned in a way similar to hash tables, but instead of bins, they have peers. This distributed data-structure is self-maintaining, self-organizing and guarantees lookups and insertions using, in most cases, $O(log(N))$ messages, where N is the number of peers in the DHT overlay [6].

DHTs are well suited for implementing a large, global index, maintained by the participants of the system. This is, in fact, straightforward: for every element of the descriptor, peers insert in the index a $\langle descriptor,\ peer{-}address \rangle$ pair. For querying, seekers make one search for each of the query descriptors to retrieve the relevant providers.

The performance gain aside, using a DHT to maintain the index shifts the responsibility from one organization to multiple organisations, which is both a

blessing and a curse: the index is no longer held in one location, meaning that no single entity has complete control of the entire index. On the other hand, the fact that it is partitioned means that many entities have control over *parts* of it. Obviously this has consequences for privacy.

Hiding the descriptors Since we do not trust the index nodes, we cannot index the content descriptors in plaintext. We identify some methods to reduce the information conveyed in the indexed descriptors and make them non-understandable to prevent direct association of content and query descriptors to users or locations.

Obfuscation Instead of indexing the descriptor itself, we index its secure hash (e.g. using SHA-1) using hash function h. For example, for the descriptor $\langle \text{“}Soccer\text{”} \rangle = 134.23.32.2$, we will store $\langle h(\text{“}Soccer\text{”}) \rangle = 134.32.42.2$. When querying, we use the same hash function for the query descriptors. Note that secure hashes are a one-way function, so it is very difficult to retrieve the value that was hashed to produce a certain result.

Approximation It is possible to perform a dictionary attack by calculating the hashes of all descriptors the attacker is interested in and matching them to the indexed descriptors. To alleviate this problem, we introduce false positives in our results. Let $P(t)$ be the set of providers that actually have content with descriptor t and $P'(t)$ the false positives. We define exposure for an index entry with descriptor t as the ratio between the number of true positives relative to the total number of answers $e_t = \frac{|P(t)|}{|P'(t)|+|P(t)|}$, a low e meaning that the providers for t have low exposure (high privacy) and $e = 1$ that they are fully exposed.

This is implemented by varying the size of the hash, so as to increase collisions. In previous work [5], we have shown that decreasing the length of the hash leads to a proportional increase in privacy. If we truncate hash values to l bits, any query would match a portion of $r = 1/2^l$ of all descriptors in the system. This is directly translatable to an average exposure of

$$e = \frac{1}{2^l} \tag{1}$$

For maximum privacy, l will be equal to 0, i.e. any query would match all descriptors stored in the index and return all peers in the system. For maximum performance, 2^l should be much higher than the total number of descriptors in the system, so as to completely avoid collisions. In [5]we have also shown that using an SHA-1 secure hash we can guarantee the privacy of individual terms for a typical webpage description corpus.

Anonymising network Hiding the descriptors in the index is not enough: identities and locations can be used to perform association attacks (see table 1 and section 5). To provide stronger security guarantees, it is desirable to hide the identity of the peers in the network. Onion routing emerged from this general

wish [2]. Onion routing anonymises communication channel and protects the identities and locations of the participants.

In onion routing, a set of nodes forms an anonymising routing overlay. The key principle, as described by Chaum[2], is that every router over which a packet is send is only aware of the identity of the previous router and the next router. This ensures that a single trusted router is in essence enough to protect the total path. This is achieved using layered encryption. Starting from the destination router, the peer will add the address of the router A_n to the data, and encrypt it with the public key K^+_{n-1} of the router before that in the routing chain. The resulting package is an onion containing several layers of encrypted addresses and data which at each level can only be read by the router with the correct private key:

$$\left[A_1, \left[A_2, [P]_{K_2^+}\right]_{K_1^+}\right]_{K_0^+}$$

When a router receives a packet, it will decrypt it with its own private key, and use the now readable address of the next router to send the contained (encrypted) data part to the next node. The destination will find the original payload when decrypting the last layer. The simplified scheme described here has been extended with nonces, symmetric keys and other mechanisms to properly ensure robust and scalable anonymity[3].

The original onion routing proposal defined 'return envelopes' to protect the identity of the client from the server. However, this method does not scale well. Next generation implementations provide other means to publish anonymously, e.g. the *Hidden Services* in the TOR project, which hide the identities both of the server and the client.

We can use this mechanism to provide volatile peer identities, i.e. *pseudonyms*. Instead of publishing $\langle descriptor, ip{-}address\rangle$ pairs on the index, it is now possible to publish $\langle descriptor, pseudonym\rangle$ pairs. After a seeker has obtained a psuedonym, it will use the anonymising network to contact the provider to retrieve the content anonymously. This will protect the privacy of the provider by making ***association*** of content to a ***real-world identity*** or ***location impossible***.

In order to protect the descriptor from statistical correlation attacks, it is possible for a provider to publish using multiple pseudonyms. We will expand on this in section 5.

5 Analysis

Here we define four classes of settings based on different combinations of the techniques discussed above.

1. ***Minimal*** A DHT is used as an index. Keys are extracted from content and queries. They are published unmodified and associated with the (IP) address of the information provider. This setting protects associations with the content and the query, since only their descriptions are published or queried.

2. ***Obfuscate and approximate*** The key is obfuscated using a secure hash function. The level of privacy and the performance impact are inversely proportional to the length of the hash l. This setting addtionally protects query and content descriptors.
3. ***Anonymise*** The key is published unmodified but the identity of the information provider and the seeker is concealed using an anonymising network. Instead of a unique address, a pseudonym to contact the peer is published. The level of privacy can be chosen by the provider at indexing time, in the choice of how many terms are connected to a single service descriptor. Compared to setting 1, this setting additionally protects associations with identities or locations.
4. ***Full*** The key is obfuscated and an anonymising network is used. This setting offers the highest privacy guarantees. Both the level of obfuscation and anonymity can be chosen. This settings protects against all associations in table 1, except for the association that some unknown user is using the system and the association that the system is used by some location. Even though it is not significant privacy-wise, even these last associations could be prevented by using a public anonymising network (e.g. [3])

Attacks We analyse the susceptibility of the aforementioned approaches to a set of common attacks. We use table 1 as grounding to describe the consequences of successful attacks.

A1: Compromised index An adversary can seize control of the index. In our approach, this is possible by compromising the DHT (how this is possible is implementation-dependent and beyond the scope of this paper). The privacy implications for a read-compromised index are negligible: the data in the index was already readable for everyone in the first place, thus no extra information is gained by taking over one or more index nodes. All settings are equally susceptible to compromising the DHT. For the rest of our analysis, we will assume that the DHT is read-compromised, i.e. that the full index is readable by the adversary.

A2: Compromised anonymiser Anonymity is provided only while the anonymising overlay is not compromised. Low-latency anonymisers are highly desirable for performance reasons, however they generally are susceptible to end-to-end communication correlation attacks [3]. Nevertheless, in order to perform this attack, the adversary needs to be able to eavesdrop on a large part of the network. Generally speaking, any compromized communication channel in setting 3 and 4 reduces the privacy for the peer(s) involved in that communication to the same level as in setting 1 and 2 respectively. The degradation of privacy is gradual and dependent on the power of the adversary to observe communication channels and the size of the system. Approaches exist to reduce these risks [3].

A3: Taking the intersection of the returned providers for correlated descriptors An attacker may exploit the fact that descriptors from the same provider are correlated. By posting several related queries and taking the intersection of the sets of returned providers, the attacker can associate provider locations or identities with descriptors.

For a set of n queries with one descriptor $t_1 ... t_n$ each , the intersection of the sets of peers returned by the system is:

$$\cap_{i=1}^{n} P^*(t_i) = \cap_{i=1}^{n} (P'(t_i) \cup P(t_i)) =$$
$$[P(t_1) \cap P(t_2) ... \cap P(t_n)] \cup \underbrace{[P(t_1) \cap P(t_2) ... \cap P'(t_n)] \cup ... \cup [P'(t_1) \cap P'(t_2) ... \cap P'(t_n)]}_{2^n - 1}$$

the set $P(t_1) \cap P(t_2) ... \cap P(t_n)$ represents all true positives for the intersection, while the $2^n - 1$ conjunctions represent the false positives, since each conjunction contains at least one P'. From previously given definition of exposure (formula 1), we have that:

$$e = \frac{|P(t_1) \cap P(t_2) ... \cap P(t_n)|}{|[P(t_1) \cap P(t_2) ... \cap P(t_n)] \cup ... \cup [P'(t_1) \cap P'(t_2) ... \cap P'(t_n)]|}$$

We will consider the worst-case scenario where the adversary somehow knows that descriptors $t_1 ... t_n$ are completely correlated in the index, i.e. they appear exactly on the same providers. Thus, $P(t_1) = P(t_2) = ... = P(t_n)$. In this case, $P(t_1) \cap P(t_2) ... \cap P(t_n) = P(t_1)$ and

$$e = \frac{|P(t_1)|}{|[P(t_1)] \cup [P(t_1) \cap P'(t_2)] \cup ... \cup [P'(t_1) \cap P'(t_2) ... \cap P'(t_n)]|}$$

In the denominator, all conjunctions with $P(t_1)$ will be absorbed by the disjunction with $P(t_1)$. Thus, $e = \frac{|P(t_1)|}{|P(t_1) \cup [P'(t_1) \cap P'(t_2) ... \cap P'(t_n)]|}$. We also have that $P(t) \cap P'(t) = \emptyset$, thus

$$e = \frac{|P(t_1)|}{|P(t_1)| + |P'(t_1) \cap P'(t_2) ... \cap P'(t_n)|}$$

The attacker has no control over the false positives $P'(t_1) \cap P'(t_2) ... \cap P'(t_n)$, which depend only on the properties of the dataset. Nevertheless, since $A \supseteq A \cap B$, the size $|P'(t_1) \cap P'(t_2) ... \cap P'(t_n)|$ will decrease as the number of conjunctive terms increases. On the other hand, it becomes increasingly difficult to find a larger set of correlated descriptors.

Note that this attack is not possible in settings 3 and 4 since the providers are using pseudonyms. It is enough to publish each correlated descriptor under a different pseudonym to leave the adversary with no additional information about the set $P(t_1) \cap P(t_2) ... \cap P(t_n)$. If it is not possible for providers to know which terms are correlated, they can publish every descriptor under a different pseudonym.

The vulnerability of setting 2 depends on the length of the hash l (since the ratio of true positives $P(t)$to false positives $P'(t)$ is dependent on l).

A4: Malicious provider tries to profile seekers A compromised index may collude with a provider to profile seekers based on their searches. Similar to the previously described attack, but profiling using queries instead of content thus attacking the privacy of the seeker instead of the provider. Seekers may prevent this by using several index nodes.

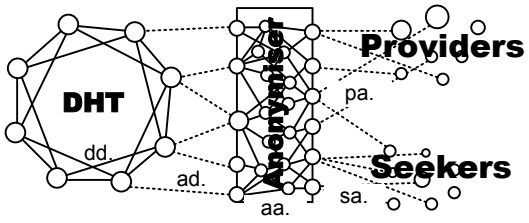

Fig. 1. Communication channels

A5: Censorship and denial of service by index nodes Though strictly speaking not a privacy attack, censorship and denial of service are related attacks that can be prevented by providing peer privacy.

A malicious index node may *censor* all descriptors that correspond to a given descriptor, by not returning any answer to queries for those. In settings 1 and 3, this is easy for any subject, since the descriptors are stored in plaintext. In settings 2 and 4, this is more difficult: since multiple descriptors map to the same hash values, the malicious node would have to censor all of them, since there is no way of knowing which ones refer to the given descriptor. The number of censored descriptors would be $1/2^l \cdot N$, where l is the length of the hash and N the total number of descriptors in the system. Detecting such a malicious index node is easy: it is enough to create some honeypot providers advertising descriptors that are likely to be censored and match results returned by the suspect index node.

Denying services to a given seeker or provider is possible in settings 1 and 2: a seeker could be presented with wrong or limited results, and the index could refuse storing the descriptors for a given provider. In settings 3 and 4, this is not possible, since providers use pseudonyms and seekers are anonymous. Assuming that malicious nodes also deny service to particular provider pseudonyms, the level of protection depends on the number of pseudonyms per provider.

Performance We will analyse the performance of our architecture focusing on the trade-off between privacy and performance. Furthermore, we will focus on its scalability in terms of network size, index size and the throughput of the anonymiser.

In figure 1, we can see the communication channels in our system. Note that the circles represent *roles* fulfilled by a physical host. E.g. a single physical host may be part of the DHT, part of the anonymising network and a content provider. In fact, we will assume that the number of hosts participating in the DHT and the anonymiser increases linearly with the number of providers and seekers.

We have the following communication channels: *dd* and *aa* for communication among DHT nodes and anonymiser nodes respectively, *ad* for DHT nodes and anonymiser nodes, *sa* for seekers and anonymiser nodes and *pa* for providers and anonymiser nodes. The cost associated with sending one message through

a communication channel is defined as $c_{ChannelName}$. E.g. the cost of doing a DHT search is c_{dd}.

Considering that DHTs and anonymiser networks can have an arbitrary number of contact points (or entry nodes), channels ad, pa and sa will not be congested. The probable scalability issue may lie within the DHT network maintaining the index(dd) or the anonymiser network(aa).

The communication resulting from a query with x descriptors is as follows: The seeker will divide it to x sub-queries with one descriptor each. Each of these will have to be routed through the anonymiser to reach the DHT, where a lookup will take place. The DHT will return s providers through the anonymiser, with a ratio r of false positives. The protocol the seeker will use to negotiate with the providers is beyond the scope of our approach. We will assume that negotiation has a cost of c_n.

Thus, the cost associated with the query is:

$$c_q = x \cdot (c_{sa} + c_{aa} + c_{ad} + c_{dd} + c_{aa} + c_{sa}) + s \cdot c_n \cdot (c_{sa} + c_{pa})$$

A typical DHT lookup cost, in terms of IP messages is $O(c_{dd}) = O(\log(N_{DHT}))$, where N_{DHT} is the number of nodes in the DHT and a typical anonymiser routing cost is $O(c_{aa}) = O(1)$. Sending messages over channels sa, ad and pa has a cost of one, since they are directly on top of the underlying network protocol. From these, we can bound the total cost of a query to:

$O(c_q) = O(x \cdot (c_{sa} + c_{aa} + c_{ad} + c_{dd} + c_{aa} + c_{sa}) + s \cdot c_n \cdot (c_{sa} + c_{pa})) = x \cdot [2 \cdot O(1) + 2 \cdot O(1) + O(\log(N_{DHT}))] + 2 \cdot s \cdot c_n \cdot O(1) = O(x \cdot \log(N_{DHT}) + 2 \cdot s \cdot c_n)$

From the definition of exposure and equation 1, we have that $e = \frac{a}{s} = \frac{1}{2^l}$, thus, $s = a \cdot 2^l$ where a is the average number of providers that match a query(true positives). We consider the cost of the negotiation constant, thus $O(c_n) = O(1)$. Then, an upper bound for query cost is:

$$O(c_q) = O(x \cdot \log N_{DHT} + a \cdot 2^{l+1})$$

where x is the number of descriptors in the query, N_{DHT} is the number of nodes in the DHT, a is the number of providers that match the query and l is the length of the hash.

It is interesting to note that the cost for using the anonymiser does not increase the overall complexity for querying. Moreover, the size of the DHT, and thus the size of the index, increase only logarithmically, indicating good scalability. The size of the hash l also plays an important role, signifying a trade-off between privacy and scalability. The limiting factor in our system is the number of matching peers a, since it is expected to grow linearly with the number of providers in the system. This makes the need for ranking in the index evident, but we will have to leave this for future work.

The cost of indexing consists of merely indexing every descriptor using the anonymiser. Due to space restrictions, we only present the final result:

$$O(c_i) = O(x \cdot \log N_{DHT})$$

where x is the number of descriptors and N_{DHT} the number of nodes in the DHT. This clearly scales well.

6 Conclusions

As pointed out in [7], privacy control has moved well beyond settings where an access control list or group access control would apply. Parties are no longer known in advance and privacy policies and trust have to be calculated on-the-fly. In an open (Semantic) Web, information sharing consists of *locating* and *acquiring* the data. Most privacy control approaches have focused on the latter. Our approach focuses on the former, providing an open discovery infrastructure where providers and queriers reveal minimal information. We have shown that our design is scalable in terms of system size and privacy level provided and that associations between aspects of content and identity can be protected. Furthermore, we have explored the trade-off between privacy and performance and analysed the message complexity for querying. A general recommendation about which setting to choose cannot be given, since it depends entirely on the requirements of an application and costs involved.

Future work includes implementation of this approach as a Sesame plug-in and integration of privacy policies.

We would like to thank Eyal Oren for the fruitful discussions and pointers and Frank van Harmelen for reviewing our work. This work is supported by the European Commission under the LarKC project (FP7-215535).

References

1. Mayank Bawa, Roberto J. Bayardo Jr., and Rakesh Agrawal. Privacy-preserving indexing of documents on the network. In *In Proc. of the VLDB*, pages 922–933, 2003.
2. David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 4(2), February 1981.
3. Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *In Proceedings of the 13th USENIX Security Symposium*, pages 303–320, 2004.
4. G. Kan. Gnutella. In A. Oram, editor, *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*, pages 94–122. O'Reilly and Associates, 2001.
5. Spyros Kotoulas and Ronny Siebes. Scalable discovery of private resources. In *IEEE SECOVAL at SECURECOMM*, Nice, France, 2007.
6. Keong Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim. A survey and comparison of peer-to-peer overlay network schemes. *IEEE Communications Surveys & Tutorials*, pages 72–93, 2004.
7. Alexandre Passant, Philipp Kärger, Michael Hausenblas, Daniel Olmedilla, Axel Polleres, and Stefan Decker. Enabling trust and privacy on the social web. In *W3C Workshop on the Future of Social Networking*, Barcelona, Spain, January 2009.
8. Sergej Zerr, Elena Demidova, Daniel Olmedilla, Wolfgang Nejdl, Marianne Winslett, and Soumyadeb Mitra. Zerber: r-confidential indexing for distributed documents. In *11th International Conference on Extending Database Technology (EDBT 2008)*, volume 261 of *ACM International Conference Proceeding Series*, pages 287–298, Nantes,France, March 2008. ACM.
9. Sergej Zerr, Daniel Olmedilla, Wolfgang Nejdl, and Wolf Siberski. Zerber+r: Top-k retrieval from a confidential index. In *12th International Conference on Extending Database Technology (EDBT/ICDT 2009)*, ACM International Conference Proceeding Series, Saint-Petersburg, Russia, March 2009. ACM.

Privacy Concerns of FOAF-Based Linked Data

Peyman Nasirifard, Michael Hausenblas and Stefan Decker

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

Abstract. In this position paper, we introduce a potential problem that arises with the emergence of publicly-available, FOAF-based linked data. The problem allows a spammer to send context-aware spam, which has a high click-through rate. Unlike online profiles within social networks, FOAF-based structured data provides a more reliable and accessible "food" for spammers and attackers. Current solutions (e.g. Digital Signatures) and proposed methods to restrict unauthorized accesses to FOAF files can prevent a subset of such activities; however we show that they are not widely used. Moreover, some of these solutions may be contrary to the mission of the Semantic Web and open data initiative.

1 Introduction

"Congratulations! You have won the National Lottery!" is a common subject line in unwanted emails (i.e. spam), which we receive in our inboxes or junk folders. Although we delete them or mark them as spam, further spam emails sometimes arrive. Key industry players (e.g. Microsoft) invest a huge amount of money in fighting spam and spammers[1], but it seems that the latter is the winner.

Various techniques have been initiated and developed for spam fighting. Labels that are identifiable by humans (i.e. CAPTCHA[2]) are currently used by major email providers to restrict the sending of automated spam. Services like tinymail[3], which aims to hide email addresses, or Email Icon Generator[4], which creates an image out of an email address, are samples of such efforts for fighting spam. Some spammers hire people to circumvent these techniques[5]. The unwritten rule of the game between spammer and "spammee[6]" is the less information we provide on the Web regarding our contact information and personalities, the lower the probability that we will receive spam.

[1] http://www.cbsnews.com/stories/2004/01/24/tech/main595595.shtml
[2] http://www.captcha.net/
[3] http://tinymail.me/
[4] http://services.nexodyne.com/email/
[5] http://www.ibm.com/developerworks/web/library/wa-realweb10/
[6] We define "spammee" as a person, who receives spam

Context-aware spam, unlike common spam, has a high click-through rate, as it contains more relevant information. This issue can be easily (ab)used by a spammer or an attacker for sharing phishing links and/or other malware. Brown et al. [1] studied the vulnerabilities of major social networks (e.g. Facebook) against context-aware spam. They estimated that around 85% of Facebook users could be accurately targeted with context-aware spam.

FOAF[7] profiles are used by many people, including Semantic Web researchers and professionals, as a means to structure contact information and social networks. FOAF profiles are considered to be "machine-interpretable", as they are based on a formal model. FOAF-based linked data helps towards enabling the mission of the Semantic Web and/or Global Giant Graph[8].

In this position paper, we describe a potential (privacy) problem of publicly-available, FOAF-based linked data. We argue that a spammer can potentially benefit from FOAF profiles by sending context-aware spam and we support this argument by presenting an example of such an attack. We believe that FOAF profiles provide a ready input for spammers, who may utilize them to personalize the spam message and increase the click-through rate of the emails. We also discuss the possible *partial* solutions that currently exist, but are not widely used.

2 FOAF: Structured Data for Spammers

FOAF profiles are structured, decentralized and extensible. They are probably the most explicit and true representation of our social networks, as people we know are clearly listed, and our contact information and interests are disclosed. FOAF profiles are an honest representation of a person's attributes, as the user takes into account the fact that they will probably only be used by machines (psychological effect). Unlike automatically-generated FOAF files[9], manually-generated FOAF files are usually hosted on personal homepages. Manually updating FOAF profiles is a time-consuming and error-prone task. Therefore, some tools have been developed to help users to create / update their profiles (e.g. FOAF-a-Matic[10]).

This structured and honest knowledge about a person's life is what spammers and/or attackers are looking for. In the following section, we demonstrate how a context-aware spam message can be constructed and sent, using a FOAF profile plus publicly available search engines (e.g. Google). Although we performed the process manually, it can be automated using available APIs and packages and/or simple text parsing.

As FOAF profiles are linked data, we need to choose some seeds which are well-connected. In our example, we used the FOAF file of "Axel Polleres[11]", as it is

[7] http://www.foaf-project.org/

[8] http://dig.csail.mit.edu/breadcrumbs/node/215

[9] As an example, FOAF profiles on http://www.deri.ie/about/team/ were generated automatically using a script

[10] http://www.ldodds.com/foaf/foaf-a-matic

[11] We had his permissions to use his FOAF profile

complete and error-free. Our goal is to send a context-aware spam message to Axel using his FOAF profile.

Figure 1 demonstrates the overall view of the process. First step is to use Google to find his FOAF file to use as a seed. Using Google API and/or parsing the HTML results can automate this process. Moreover, using some available techniques [2], such as utilizing the filetype option, can help us to reach the desired FOAF profile at first or higher ranks (i.e. *filetype:rdf*). Our experiment with the query "Axel Polleres FOAF" returned the requested FOAF file as the first retrieved link. We followed the link and accessed Axel's FOAF profile. As it is well-structured linked data, it can be parsed using common RDF processing libraries, like Sesame or Jena. Parsing Axel's FOAF profile gave us valuable information about his friends and contact information.

The next step is to find the seed's email address. FOAF profiles include SHA1 hash code of email addresses. As SHA1 collisions are far from current power of computers[12], they are good sources for "verification" purposes. Therefore, for finding email addresses, we follow a similar approach to that for finding FOAF profiles. Our experiment with Google using query "Axel Polleres Email" returned the requested email as the first retrieved link. However, the result is a HTML page and needs further processing, but having the SHA1 hash code of the email, is used as a help. Many people, including our seed, try to mask their email addresses using various text-based techniques such as replacing "@" with "[at]" etc. After gaining access to a HTML page that contains the seed's email address, we first search for the keyword "email" or "e-mail" on that page. We then apply the following common patterns to retrieve the email:

- Remove spaces
- Replace 'at' or '[at]' with '@', and 'dot' or '[dot]' with a period
- Replace 'firstname' with the user's actual first name, and 'lastname' with the user's actual last name (gleaned from FOAF profile)
- Remove 'removeme', '[removeme]'
- other possible rules

As generating a SHA1 hash code is fast, we may continuously check if we have a valid email address. If we did not succeed at the first retrieved link, we may refer to the second or third results of search engine. Note that some people generate SHA1 hash code of their emails without the "mailto:" prefix.

Using these techniques, we can successfully identify Axel's email address. In order to send a context-aware spam message, we need to identify Axel's friends and their contact information as well. Finding Axel's friends from his FOAF profile is straightforward, as they are clearly listed in the profile. In order to find their emails, we repeat the previous method.

One may claim that a possible problem that arises from this approach is resolving name ambiguity. As emails are unique and we have access to the SHA1 of the email, we always ensure the hash code of a person's email matches the SHA1. This will automatically resolve name ambiguity issues. Obviously, this may require parsing more results from the search results.

Returning to the scenario, we now have the email address of the seed plus the email addresses of his friends. In order to send a context-aware spam, we may use

[12] http://www.schneier.com/blog/archives/2005/02/cryptanalysis_o.html

some pre-defined templates. Based on the granularity of information that people provide in their FOAF profiles, we select the appropriate template.

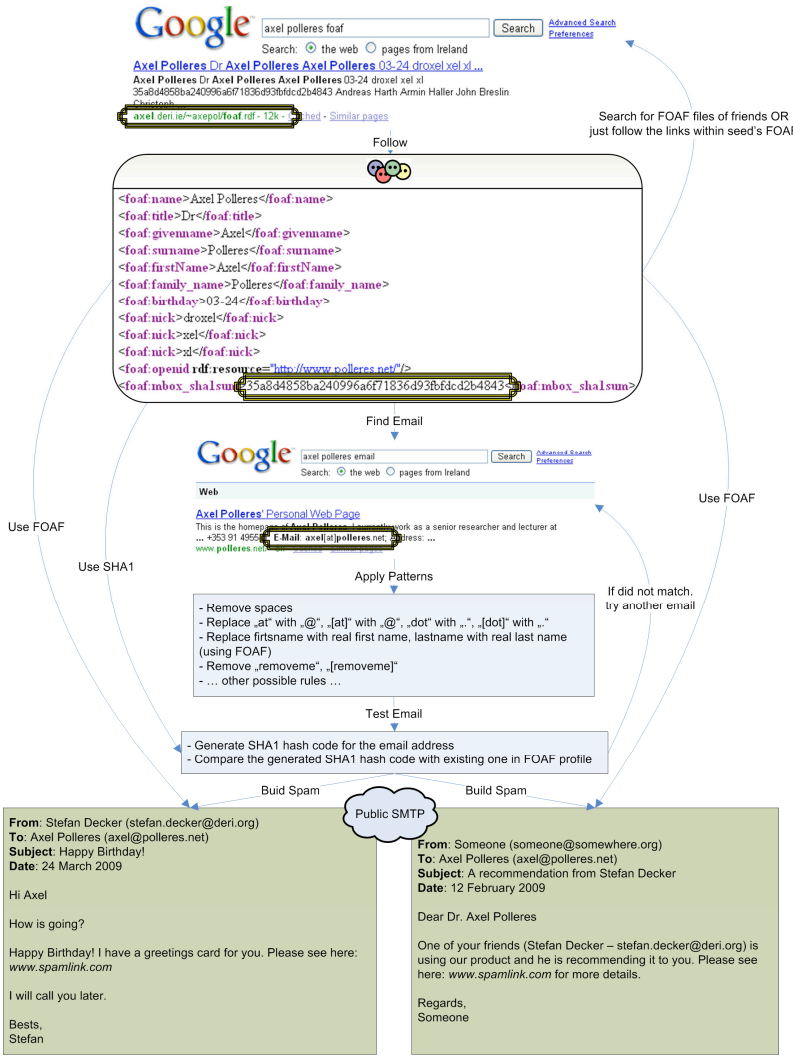

Fig. 1. Overall view of sending spam using FOAF

For example, Axel put his birth date in his FOAF profile. A context-aware spam may be sent on his birth date from one (or more) of his friends, which encourages him to visit an online greetings card, that contains some harmful/harmless links. Another template may use Axel's friends for advertisement purposes. Figure 1 demonstrates

two sample spam messages that have been created using some templates: one for a birthday and the other for sending a product recommendation. Using techniques and tools like URL shortening[13] can potentially be used to further hide the content and target of the links.

For our experiment, we did not use a template. We created a simple email with an embedded link and sent it via a customizable SMTP server to our victim (seed). He clicked that link, as it seemed to be from one of his friends.

This approach is recursive. The seed's friends may be used as new seeds and the process continued. Taking into account the small world phenomenon (i.e. Human Web), we can expect to reach all people in the FOAF-o-sphere. If not, common (Semantic) search engines can be used for identifying new seeds. We can also consider mutual links between people in FOAF profiles may represent a stronger / more reliable relationship, which can in turn be used for increasing spam's click-through rate.

3 Discussions and Potential Solutions

Privacy of FOAF profiles has been discussed by some researchers. Reagle [3] proposed to encrypt parts of the FOAF file to restrict unauthorized accesses. Frivolt and Bieliková [4] proposed to partition FOAF files, where each partition has a specific visibility. These solutions are not widely used and FOAF files are accessible to all and are indexed by major (Semantic) search engines. Moreover, it seems that hiding FOAF profiles can be contradictory to the open data initiative.

In comparison to common social networking sites, using FOAF for sending context-aware spam can be considered as a more reliable and accessible approach for spammers, due to several reasons:

- Finding users' email from online social networks could be very difficult, as most social networking sites hide the email addresses of the users. SHA1 hash code of the emails within FOAF profiles can be used as a means for verifying the valid email address.
- Crawling heterogeneous and highly customizable social networks (e.g. MySpace) offers a huge overhead for spammers, whereas FOAF is unique structured data.
- Someone may generate fake user profiles with incomplete names within online social networks, whereas FOAF is considered to be "reliable", as they are hosted on personal homepages and/or automatically generated from reliable data. Although, we are not aware of any study on the degree of "honesty" of FOAF profiles.

Generally, generated context-aware spam using FOAF can be categorized into two main groups: the first is where the sender of the email is supposedly a friend of the seed, then second is where the sender of the email is unknown to the seed, however the email may refer to friend(s) of the seed. Figure 1 illustrates examples of both categories.

[13] http://tinyurl.com/

Digital Signatures can potentially obstruct spam of the first type, but they are not widely used. The result of a survey that we did within our institute showed that just one person out of one hundred and twenty researchers and staff is using digital signatures permanently. Three researchers claimed that they use it, whenever they want to look serious or they want to contact somebody officially. One participant claimed that he used to have digital signature, but as his friends do not use it, he found it useless and he stopped using it. One participant claimed that he always wanted to have one, but due to time limitations, he did not investigate how he can install it on his email client. One participant claimed that managing private key / public key is very time-consuming. He said, however, there exist some tools that can help towards key management, but they require user verification (i.e. human-in-the-loop) at the end which brings lots of overhead for users. One participant with sufficient technical background tried to install a certificate on his email client, but after half an hour struggling, he stopped and complained about its complexity and argued that the process can be very time-consuming for a common user with no or little technical background. The result of the survey showed that 114 out of 120 participants never used digital signature within their email clients. The fact that lots of people are not really using digital signatures undermines the usefulness of digital signatures for those, who are using them permanently.

The other possible solution for the first group of spam is looking at detailed headers of the emails. This will show the "traversed" path of an email through its journey to reach the target person. This is probably a technical point which many users are not aware of and even for technical users, it is a time-consuming issue. Moreover, there exist some RFCs (e.g. RFC 4408 – Sender Policy Framework (SPF)) that help towards this direction, but they are not widely used.

For the second group of spam, spammers may always use major and free email providers for sending spam. This is still an open problem. Thanks to publicly available FOAF profiles, spammers can increase click-through rate of the spam by making them context-aware.

Generally, some partial solutions can be also considered to increase privacy of FOAF profiles:

- Remove SHA1 hash code from FOAF
- Use various hashing functions within FOAF, and not only SHA1
- Mask person's name and/or friends' name within FOAF

4 Conclusion

We all receive spam. Although FOAF-based linked data can bring lots of advantages for the community, it is necessary to be aware of malicious usage of such data. In this paper, we presented a potential privacy leak of publicly available FOAF profiles, demonstrated a context-aware spam that was sent just by using FOAF files, search engines and a customizable SMTP server and our victim clicked on the embedded link. We also argued that current *partial* solutions (e.g. digital signatures) are not widely used and if we put our FOAF profiles online, we may expect context-aware spam.

Acknowledgement. We thank anonymous reviewers for their feedbacks. We also thank all people, who participated in our survey. The work presented in this paper has been funded in part by SFI under Grant No. SFI/08/CE/I1380 (Lion-2) and the EU under Grant No. FP6-IST-5-35208 (Ecospace).

References

1. Brown, G., Howe, T., Ihbe, M., Prakash, A., Borders, K.: Social networks and context-aware spam, in Proceedings of the ACM conference on Computer supported cooperative work. ACM, San Diego, CA, USA (2008)
2. Calishain, T., Dornfest, R.: Google Hacks: 100 Industrial-Strength Tips and Tools. O'Reilly and Associates, Inc. 352 (2003)
3. Reagle, J.: FOAF Spheres of Privacy. Available online at http://reagle.org/joseph/2003/09/foaf-spheres.html
4. Frivolt, G., Bieliková, M.: Ensuring privacy in FOAF profiles. in Znalosti (2007)

Data Provenance on the Social and Semantic Web

Trust and Privacy on the Social and Semantic Web (SPOT 2009)

Data Republishing on the Social Semantic Web

Claudia Wagner[1,2] and Enrico Motta[2]

[1] Institute for Networked Media, JOANNEUM RESEARCH,
Steyrergasse 17, 8010 Graz, Austria
claudia.wagner@joanneum.at
[2] Knowledge Media Institute, The Open University,
Walton Hall, Milton Keynes, MK7 6AA, United Kingdom
e.motta@open.ac.uk

Abstract. Data Republishing is a recent Social Web phenomenon which can be observed in different areas of the Social Web. However, current Data Republishing tools don't work in the emerging context of the Semantic Web. In particular, these tools neither generate any semantic metadata which provide information about the republished content (e.g., provenance information) nor are they able to make use of existing semantic metadata annotating the original content being republished. In this work we introduce the concept of Semantic Data Republishing and describe how to implement it.

1 Introduction

1.1 Motivation

Data Republishing is a recent Social Web phenomenon which can be observed in different areas of the Social Web, such as the blogosphere, the microblogosphere or the social networking sphere. Data Republishing refers to the process in which a user, knowing that data are already published on the Web, rereleases them in a new context. Users for example republish data by *reblogging* external content on their blogs, by *retweeting* microblog posts from other users on their own microblog or by *posting* external content to their Facebook[3] wall. A new kind of republishing oriented Social Web application, so-called tumblelogs, has recently emerged from this trend. Tumblelogs are blogs with shorter posts and mixed media types which are usually less structured than classical blogs [19]. Users can quickly share their online discoveries by republishing multimedia content, found on the Web, on their tumblelogs. Tumblelog providers, such as tumblr[4] and soup[5], gain in importance thanks to their increasing number of unique visitors[6].

[3] http://facebook.com
[4] http://tumblr.com
[5] http://soup.io
[6] http://siteanalytics.compete.com/soup.io+tumblr.com/?metric=uv

Current Data Republishing tools, such as Tumblr Share[7] , ShareThis[8] or Zemanta Reblog[9], support users in republishing their online discoveries on Social Web applications. These tools allow users to select data on any web page, generate a new data item on their preferred target web application, transfer the selected data as text or binary data and use them as content of the new data item. However, a limitation of this approach is that no semantic metadata are generated - e.g., to expose the provenance of the copied data. That means that the information about the republishing process (i.e., who republished, when, from which source application, which fragments of data on which target application) is lost. Another drawback of current Republishing tools is that they are not able to make use of existing semantic metadata which may annotate the original data being republished. Consequently, these tools do not fully support the next generation of Social Web applications, so-called Social Semantic Web applications, which expose the semantics of their data in a machine-interpretable way by using ontology-based metadata.

In this paper we illustrate the need of a new kind of Republishing tool for the Social Semantic Web. We introduce the concept of Semantic Data Republishing and discuss requirements and functionalities of tools implementing this concept in section 2. An initial implementation of an example prototype implementing Semantic Data Republishing is presented in section 3. Finally, in section 4 and 5 we discuss related work stemming from the areas of data publishing and Data Portability on the Social Semantic Web and outline new opportunities for research and development made possible by it.

1.2 Data Republishing on the Social Semantic Web

Two different methods for Data Republishing across individual web sites can be distinguished: (1) Data Republishing by copying data values and (2) Data Republishing by copying data references.

(1) Data Mobility standards (e.g., RSS 1.0, RSS 2.0, Atom, OPML) facilitate Data Republishing by copying data values [9]. Thanks to Data Mobility initiatives structured data can be republished on individual websites without the need to implement application-specific Programming Interfaces (APIs).
(2) Linked Data Design Principles[10] provide data access by reference. Hence, data published according to these principals can be republished and reused by reference. Social Semantic Web applications can reference and dereference resources by using their URIs, access their machine-interpretable descriptions and republish data without the need to copy data values.

Both methods, i.e. Data Republishing by reference and Data Republishing by value, are important for different scenarios.

[7] `http://www.tumblr.com/goodies`
[8] `http://sharethis.com`
[9] `http://zemanta.com/reblog`
[10] `http://www.w3.org/DesignIssues/LinkedData.html`

If data are republished by copying their values both, the source and the target application, store an individual instance of the same data. These instances can then be changed individually. Therefore this technique is suited for situations in which users want their republished data to be independent of the original data (e.g., because users do not want the republished data to change, if the original ones change or because the original data may not be available for long).

If data are republished by reference, the source and the target application point to the same data instance. In this scenario, if the source or target application modifies the data, the data being displayed change on both applications. Therefore this approach ensures that in situations where data are likely to be modified (e.g., in the context of a wiki page) the republished data and the original data are kept in sync.

The advantage of exploiting Semantic Web technologies in the context of the current republishing phenomenon of the Social Web is that the two aforementioned republishing methods can be integrated to combine the advantages of both approaches. In particular by applying Semantic Web technologies to Data Republishing data can be cached on the target application to increase the availability of republished data and can in addition be updated at certain intervals by using the semantic metadata of the republished data to formulate queries.

2 The Design of a Semantic Republishing Tool

2.1 Requirements

A Semantic Republishing tool should allow users to select content from any web site and republish it on their preferred Social Web or Social Semantic Web application (e.g., their blog, their Facebook wall). To exploit the full potential of the Social Semantic Web in the context of the current Republishing trend we have identified the following main requirements for Semantic Republishing tools:

1. **Semantic Republishing tools must be able to detect and republish semantic metadata together with the data they annotate.** Semantic metadata must be republished together with the original data they annotate to allow users to benefit from additional third party services and tools which leverage semantic metadata of the data currently being processed. These additional services need semantically described structured data in order to be able to interpret the data and provide services upon them.
   For example browser tools, such as Firefox Operator[11], leverage semantic metadata found on the currently viewed web site and provide services (such as "Export contact to MS Outlook address book") upon the data which are annotated by the processed metadata.
2. **Semantic Republishing tools must be able to generate new semantic metadata exposing information about the provenance of the republished data.** If data are republished in a new context, new semantic

[11] `https://addons.mozilla.org/de/firefox/addon/4106`

metadata must be created which expose information about the provenance and the republishing process in a machine-interpretable way. Consequently, Semantic Web search engines can use this information to answer sophisticated data queries (such as *select all users who republished this section of this article* or *select all comments about a certain youtube*[12] *video related with the original video or with posts embedding the video*). Furthermore, it is important that Semantic Republishing tools expose detailed provenance metadata to boost the transparency and information accountability on the Web (see section 2.3). Finally, the exposure of detailed machine-interpretable provenance metadata allows implementing synchronization services which keep the republished data and the original ones in sync.

3. **Semantic Republishing tools must be able to interpret semantic metadata associated with the data to republish.** Existing semantic metadata can expose information about the content, the structure, the privacy settings and usage restrictions of the data they annotate. Hence, existing semantic metadata annotating the original data must be interpreted by Semantic Republishing tools in order to support users during the Republishing process (e.g., suggest tags of original data to reuse or suggest how to republish original data according to their licenses).
4. **Semantic Republishing tools must be easy to use for end-user.** To minimize usage barriers the interface of the Semantic Republishing tools must be similar to interfaces of already widely used traditional Republishing tools.

2.2 Metadata Modelling

We use the SIOC[13] ontology (namespace prefix `sioc`) together with the DCMI Metadata Terms[14] (namespace prefix `dcterms`), the Dublin Core Metadata Element Set[15] (namespace prefix `dc`), the FoaF[16] Ontology (namespace prefix `foaf`) and the RDF Site Summary 1.0 Module Content[17] (namespace prefix `content`) to describe republished data items in a machine-interpretable way. A republished data item is exposed as a resource of type `sioc:Post` and identified by a URI (e.g., `http://example.com#rebloggedItem_443af`) to enable any third party to make reference to this item in other RDF statements. The `sioc:content` property is used to expose the plain text content and the `content:encoded` property is used to expose the (X)HTML content of a republished item. The `dc:source` property relates the republished item with the resource from which it originates. The `dcterms:created` property exposes the date and time when the republished content has been published for the last time.

[12] `http://youtube.com`
[13] `http://rdfs.org/sioc/spec/`
[14] `http://dublincore.org/documents/dcmi-terms/`
[15] `http://dublincore.org/documents/dces/`
[16] `http://xmlns.com/foaf/spec/`
[17] `http://web.resource.org/rss/1.0/modules/content/`

2.3 Related Privacy and Usage Rights Issues

In the context of Data Republishing privacy and usage policies related with the data being republished must be taken into account. Privacy policies specify the confidentiality of data during transmission and also after receipt of data [10] and usage policies specify how and under which conditions clients are allowed to use data. With current widely-used Republishing tools users can either republish all data for which they have reading permissions without taking privacy and usage policies into account or cannot republish private or usage restricted data at all. Usage rights and privacy settings related with the selected data cannot be taken into account by these tools, because the settings are usually neither published in a machine-interpretable way nor are these tools able to interpret them. Consequently, traditional Data Republishing tools cannot support users in republishing data without compromising privacy and usage policies of data.

We believe that Semantic Data Republishing tools can help to overcome this problem and support privacy and right data usage by taking one of the following approaches:

(1) Interpreting privacy and usage policies related with the data being reused to guide users through the republishing process -i.e. support users in republishing data without compromising privacy policies or usage restrictions.
(2) Preserving privacy and usage policies of the original data when they are republished to enforce them for the republished data as well.

First, to allow Semantic Republishing tools interpreting policies of data web applications must expose not only their data in a machine-interpretable form, but also the related privacy and usage policies. If policy metadata are embedded in web pages to relate data with their policy descriptions, Semantic Republishing tools will be able to extract and interpret them. Consequently, users will be informed about policies related with the data they want to republish and will be warned if they are going to violate policies by republishing data. The Creative Commons Rights Expression Language (ccREL) [1], the standard recommended by Creative Commons (CC) for machine-readable expression of usage rights, is a successful example of publishing lightweight usage rights encoded in XHTML+RDFa. The proposed interpreting and guiding functionality of Semantic Republishing tools will however not prevent users from abusing data and compromising privacy and usage settings, but boost user's awareness of data privacy and 'good' data usage. This user's awareness combined with a transparent republishing process can ensure privacy through fair, appropriate and transparent use of information [20]. Semantic Republishing tools expose the provenance and republishing history of data in a machine-interpretable form. Consequently, users who violate usage and/or privacy policies related to data being republished can then be held accountable.

Second, to allow source and target application to share data and their policies Semantic Republishing tools must make the relation between the original data and the republished data explicit. Existing policy frameworks, such as REIN [11] or Protune [7], can be used on source and target applications to share the

policies of the original data and reason over them. Both frameworks are based on Semantic Web technologies and can be used for representing and processing distributed policies. However, in the context of Data Republishing the same data can be accessed on the source and target application. Therefore the source and target application must both be able to enforce the policies of the original data or the target application must redirect the client request to the source application which can consequently enforce the policies of original data for the republished data as well.

3 Implementation of a Semantic Reblog Tool

To demonstrate our ideas we have implemented a first example of a Semantic Data Republishing tool, namely a Semantic Reblog tool for the OpenSource Blogging Software WordPress[18]. The Semantic Reblog prototype consists of a client side bookmarklet and a server side reblog script. This section gives some insight into implementation issues and describes the Semantic Data Republishing process.

3.1 Extraction of semantic metadata

The Semantic Reblog tool extracts semantic metadata which annotate the current user selection (see step 1 and 2 in figure 1). On the client side the Semantic Reblog bookmarklet uses jQuery RDF plug-ins[19] to extract semantic metadata which are embedded in the selected (X)HTML region of the current web site. If no semantic metadata related with the selected (X)HTML region can be found, the Semantic Reblog server component parses the whole (X)HTML page searching for links to external RDF files which describe the page's data. The Semantic Reblog server component extracts triples from the external RDF files as well and checks if the selected data values belong to any object values of the extracted triples. The Semantic Reblog server component uses ARC2[20] to parse and extract semantic metadata. It must be noticed that the results of this server side extraction process can be ambiguous and that therefore the results of the client side extraction which takes positional information as well into account are usually more precise.

3.2 Generation of semantic metadata

The Semantic Reblog server component pastes the selected data into the tinyMCE editor[21] which is used as visual user editor by WordPress (see step 3 in figure

[18] `http://wordpress.org`
[19] `http://code.google.com/p/rdfquery`
[20] `http://arc.semsol.com`
[21] http://tinymce.moxiecode.com/

1). As described in section 2.2 the republished data are automatically annotated with semantic metadata exposing their provenance. All semantic metadata are embedded in the (X)HTML of the post's content and are serialized in XHTML+RDFa.

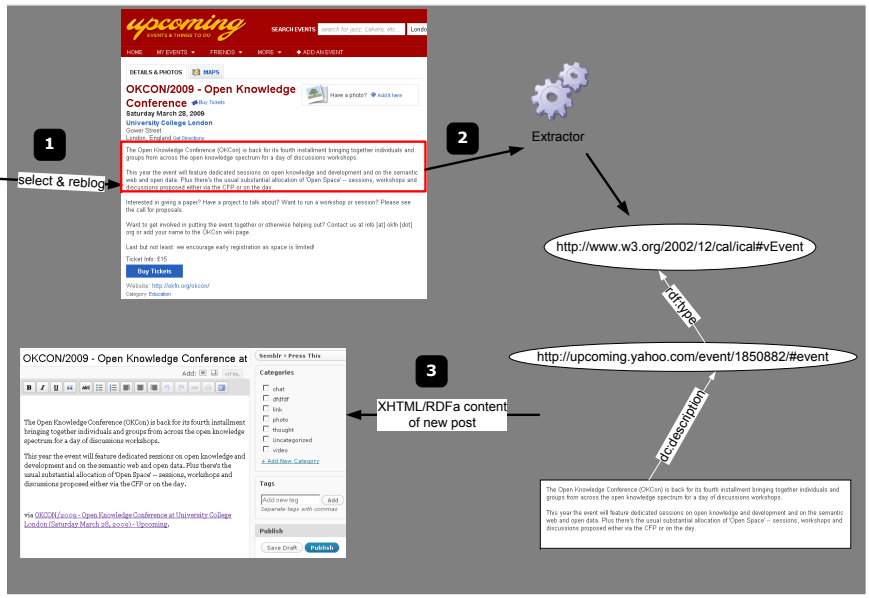

Fig. 1. Semantics-aware reblog process: extract and edit data and semantic metadata

3.3 Republishing data and semantic metadata

The Semantic Reblog tool preserves existing semantic metadata embedded in the selected content of the source site and republishes them together with the newly created semantic metadata and the data being annotated by them. Two different approaches have been identified for preserving semantic metadata embedded in (X)HTML snippets during the republishing process:

(1) **RDFa Serialization:** The RDF graph which has been extracted from the selected fragment of the source site can be serialized as XHTML+RDFa snippet. The disadvantage of this approach is that the parts of the selected (X)HTML content which are not semantically annotated get lost.

(2) **Snippet *Semantification*:** Cutting individual (X)HTML snippets from a semantically enriched (X)HTML pages can lead to (X)HTML snippets which contain meaningless, local and/or incomplete semantic metadata.

During the *semantification* process the semantic metadata embedded in the selected (X)HTML snippet are transformed into a valid semantically enriched (X)HTML snippet by reusing the semantic metadata extracted from the source site. The *semantified* (X)HTML snippets are serialized as XHTML+RDFa and are stored in a post's content.

Finally, the Semantic Reblog tool makes the republished and newly generated data and semantic metadata accessible for further web applications (see figure 2). The *semantified* (X)HTML snippet together with the newly generated semantic metadata are displayed in a user's reblog editor and can be edited by the user (see step 1 in figure 2). The editor can either be used in the visual edit mode in which the (X)HTML mark-up is hidden or in the HTML mode in which the data and their mark-up are displayed. The user can push the *publish* button to publish the post (see step 2 in figure 2). A newly created post is displayed on the user's blog. To make the embedded, reblogged resources accessible the WordPress SIOC Exporter[22] which models the content of a blog semantically and serializes it as RDF/XML document has been extended. The extended WordPress SIOC Exporter[23] is used to export resources embedded inside a post's content (e.g. reblogged data items) and relates them with the blog post via the `sioc:embeds` property of the SIOC ontology (see step 3 in figure 2) . Finally, the Semantic Web index service Sindice[24] is pinged to ensure that the republished semantically annotated data are indexed (see step 4 in figure 2).

3.4 Usage Scenario

To illustrate the benefits of our Semantic Reblog tool, an example scenario is described:

Tim is a typical Social Web users and one of his hobbies is taking pictures and sharing them on the Social Web application Flickr[25]. He likes discussing his pictures with other users interested in photography. Tim browses the Web and stumbles across one of his pictures which has been republished on the tumblelog of someone he does not know. The republished picture has been commented on the tumblelog and Tim is happy that he found such nice comments about his picture. Tim starts being interested in who else might have republished his picture. In particular, he would like to find all comments about his picture, no matter on which application they have been published. That means that Tim wants to find as well comments about postings which have republished his picture. Tim uses RepuSearch which is the fictive Semantic Web search engine specialized in querying the republishing-sphere. RepuSearch provides a simple search form allowing users to specify what they are searching for and formulates SPARQL queries in the background. Tim copies the URI of his picture into the main

[22] `http://sioc-project.org/wordpress/`
[23] `http://clauwa.info/download`
[24] `http://sindice.com/`
[25] `http://flickr.com`

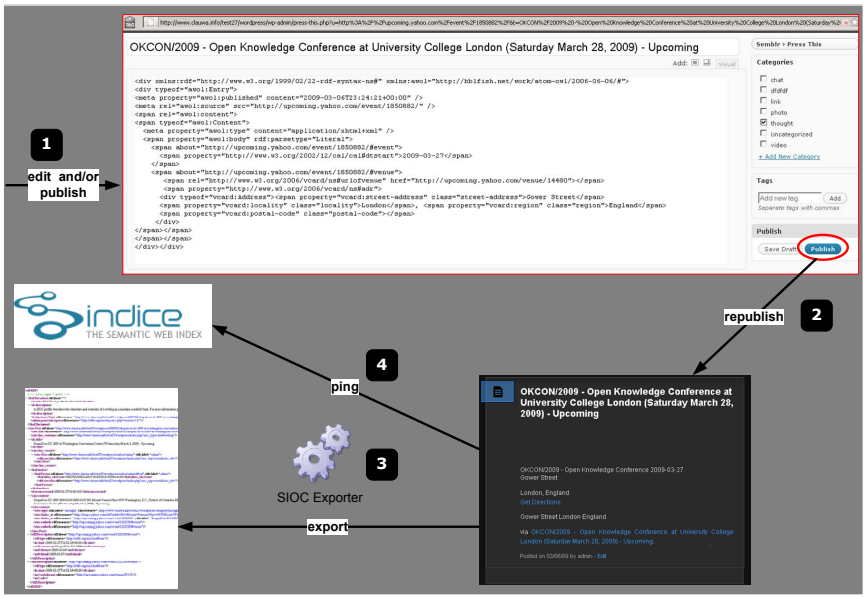

Fig. 2. Semantics-aware reblog process: republish and disseminate data and semantic metadata

search box, specifies that he wants to find comments about his picture which have been created in the last month and pushes the *search* button. RepuSearch displays as a result a list of comments created in the last month which refer to resources (e.g. posts) embedding Tim's picture.

Based on our work a scenario like this can be realized in the future Web where Semantic Web search engines exist allowing and supporting users in querying the Web like a huge database.

4 Related Work

There has been a significant amount of research in publishing and interlinking data on the Social Semantic Web. Social Semantic Web applications, such as semantic blogs [14] [12] [5] [18], semantic wikis [16] or semantic microblogs [15], allow average users to publish and interlink their data in a machine-interpretable way. Our work distinguishes from aforementioned work by focusing on republishing data. The requirements and challenges of publishing and modelling already published data slightly differ from publishing unpublished data and additional topics such as data privacy and usage rights arise in this context (see section 2.3).

SemiBlog [13] [3] illustrates how users can annotate their blog posts with existing metadata from desktop applications. SemiBlog and our prototype both focus on reusing existing semantic metadata. However, unlike semiBlog our Semantic Reblog prototype reuses semantic metadata of web resources. On the contrary semiBlog reuses metadata stored on desktop applications.

The Snippet Manager [6] and PiggyBank [8] are tools which allow users to collect, manage and share information found on the Web. Both tools are centralized services, which store the information snippets of a user in his or her personal semantic bank or knowledge base. Users can share information with other users by granting them access to parts of their knowledge base or semantic bank. On the contrary our Semantic Reblog prototype is not a centralized service, but generates semantically enhanced information snippets which are stored on distributed web applications. Furthermore, the Semantic Reblog prototype allows republishing information snippets or modified versions of them in a new context.

The work by Bojars et al. [4] shows how Semantic Web technologies can be used to ensure portability of user-specific data and content. In particular they propose to use FoaF and SIOC ontologies to model user information and user-generated content in a machine-interpretable way. Current application specific SIOC Importers and Exporters[26] demonstrate how data can be migrated from one Social Web application to another. After the portability process of a certain resource (e.g., a blog post) the target site holds a replica of the original resource. Our work distinguishes from their work by addressing another scenario in which a user does not want to generate and republish a replica of the original resource. On the contrary a user wants to generate a new resource which embeds and/or discusses the original resource or parts of it.

Semantic Clipboards aim to realize Data Portability across all kinds of applications. The Semantic Clipboard idea was first presented by [2] and describes how Semantic Web technologies can be used for moving structured content across application boundaries. The source and destination application negotiate the format of the data to be transferred and the clipboard itself either holds a copy of the RDF description of data or a reference pointing to the data's RDF graph. A first implementation of the Semantic Clipboard is presented in [17] and allows copying RDF metadata from any source application to any desktop applications. Other implementations of the Semantic Clipboard idea such as the RDFa Clipboard[27] and Semsol's Web Clipboard[28] exist as well. As the Semantic Clipboard idea aims to solve a very generic problem our work can be seen as an easy-to-use, lightweight and pragmatic solution for a specific problem in the context of the current republishing trend of the Social Web. Furthermore, Semantic Clipboards are made to act on an ideal Semantic Web where all published data are described in machine-interpretable way. Only semantically annotated data can

[26] http://rdfs.org/sioc/applications/

[27] http://www.w3.org/2006/07/SWD/RDFa/impl/js/rdfa-clipboard/

[28] http://bnode.org/blog/2006/06/12/web-clipboard-adding-liveliness-to-live-clipboard-with-erdf-json-and-sparql

be clipped and reused by using Semantic Clipboards. Our Semantic Reblog prototype however is designed to be used on the current Web where not all data are semantically annotated, but can be republished.

5 Conclusions and Future Work

In this paper we discussed the current republishing phenomenon on the Social Web, highlighted related privacy issues and illustrated the benefits of using Semantic Web technologies for Data Republishing. We introduced the concept of Semantic Data Republishing and described requirements and potentials of a new generation of semantics-aware Republishing tools. We presented a first implementation of such a tool, namely a Semantic Reblog tool, which allows users to republish data and their semantics, found on the Web, and annotates them with ontology-based metadata exposing their provenance. However, a number of issues still need to be addressed including how Semantic Republishing tools should handle the republishing of private and/or usage restricted data and how current Social Web applications can export their privacy policies in a machine-interpretable way to make them reusable for other applications. We plan to address these issues in future versions of our Semantic Reblog tool. Furthermore, we plan to improve the user interface of our Reblog tool to separate the semantic annotations from the editable (X)HTML code and hide them from the user.

References

1. Hal Abelson, Ben Adida, Mike Linksvayer, and Nathan Yergler. ccREL: The Creative Commons Rights Expression Language, March 2008.
2. Tim Berners-Lee. Semantic Clipboard. http://www.w3.org/DesignIssues/SemanticClipboard, January 2004. accessed: 12.2.2009.
3. Uldis Bojars, John G. Breslin, and Knud Möller. Using Semantics to Enhance the Blogging Experience. In *ESWC*, pages 679–696, 2006.
4. Uldis Bojars, Alexandre Passant, John G. Breslin, and Stefan Decker. Social Network and Data Portability using Semantic Web Technologies. In *2nd Workshop on Social Aspects of the Web (SAW 2008) at BIS2008*, pages 5–19, 2008.
5. Steve Cayzer. Semantic blogging and decentralized knowledge management. *Commun. ACM*, 47(12):47–52, 2004.
6. Steve Cayzer and Paolo Castagna. How to build a Snippet Manager. In Stefan Decker, Jack Park, Dennis Quan, and Leo Sauermann, editors, *Proc. of Semantic Desktop Workshop at the ISWC, Galway, Ireland, November 6*, volume 175, November 2005.
7. Juri Luca De Coi, Daniel Olmedilla, Piero A. Bonatti, and Luigi Sauro. Protune: A Framework for Semantic Web Policies. In Christian Bizer and Anupam Joshi, editors, *International Semantic Web Conference (Posters & Demos)*, volume 401 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.
8. D. Huynh, S. Mazzocchi, and D. Karger. Piggy Bank: Experience the Semantic Web inside your web browser. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(1):16–27, 2007.

9. Kingsley Idehen and Orri Erling. Linked Data Spaces & Data Portability. Linked Data on the Web workshop (LDOW2008), 2008.
10. Lalana Kagal, Tim Berners-Lee, Dan Connolly, and Daniel J. Weitzner. Using Semantic Web Technologies for Policy Management on the Web. In *AAAI*, 2006.
11. Lalana Kagal, Massimo Paolucci, Naveen Srinivasan, Grit Denker, Tim Finin, and Katia Sycara. Authorization and privacy for semantic web services. In *IEEE Intelligent Systems*, pages 50–56, 2004.
12. David R. Karger and Dennis Quan. What Would It Mean to Blog on the Semantic Web? *The Semantic Web ISWC 2004*, pages 214–228, 2004.
13. Knud Möller, John G. Breslin, and Stefan Decker. semiBlog - Semantic Publishing of Desktop Data. In *14th Conference on Information Systems Development (ISD2005), Karlstad, Sweden*, pages 855–866, Karlstad, Sweden, August 2005.
14. Ikki Ohmukai and Hideaki Takeda. Semblog: Personal Knowledge Publishing Suite. In *Proceedings of WWW 2004 Workshop on the Weblogging Ecosystem: Aggregation, Analysis and Dynamics*, New York, USA, 2004.
15. Alexandre Passant, Tuukka Hastrup, Uldis Bojars, and John Breslin. Microblogging: A Semantic Web and Distributed Approach. 2008.
16. Alexandre Passant and Philippe Laublet. Towards an Interlinked Semantic Wiki Farm. In *SemWiki*, volume 360 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.
17. Gerald Reif, Gian Marco Laube, Knud Möller, and Harald Gall. SemClip - Overcoming the Semantic Gap Between Desktop Applications. In *Semantic Web Challenge*, volume 295 of *CEUR Workshop Proceedings*, 2007.
18. Aman Shakya, Hideaki Takeda, Vilas Wuwongse, and Ikki Ohmukai. SocioBiblog: A Decentralized Platform for Sharing Bibliographic Information. In Pedro Isaas, Miguel Baptista Nunes, and Joo Barroso, editors, *Proceedings of the IADIS International Conference WWW/Internet 2007*, volume 1, pages 371–380, Vila Real, Portugal, October 2007. International Association for Development of the Information Society, IADIS Press.
19. Alexander Stocker, Johannes Müller, and Klaus Tochtermann. Leichtgewichtiges Bloggen im Umfeld von Unternehmen: Microblogs und Tumblelogs. *e-commerce*, 2009.
20. Daniel J. Weitzner, Harold Abelson, Tim B. Lee, Joan Feigenbaum, James Hendler, and Gerald J. Sussman. Information accountability. *Commun. ACM*, 51(6):82–87, 2008.

Provenance: The Missing Component of the Semantic Web for Privacy and Trust

Harry Halpin
H.Halpin@ed.ac.uk

School of Informatics
University of Edinburgh
2 Buccleuch Place
EH8 9LW Edinburgh
Scotland, UK

Abstract. Data on the Semantic Web currently does not have any standardized or any *de-facto* agreed upon way to exhibit provenance information, yet provenance is the foundation for any reasonable model of privacy and trust. Yet, currently every RDF triple does not have any coherent way of storing provenance information on the Semantic Web. We present the hypothesis that provenance is by far the most important data needed on the Semantic Web for privacy and trust, and review previous work in database systems on provenance. We put forward the concept that the three main provenances operators (insertion, deletion, and copy) from provenance work in database systems can be used on the Semantic Web. Furthermore, we hypothesize that such information naturally should be stored in or using the name URI of named graphs. We show that such an approach can help solve practical issues of privacy and trust in social networks using a real-world example.

Keywords: *provenance, trust, Semantic Web*

1 A Theory of Provenance

Provenance has remained for the most part an undeveloped area of research, both in traditional database systems and on the Semantic Web. The most well-known approach presented by Berners-Lee, Kagal, and others [1] attempts to capture provenance information in terms of proofs using a Datalog-like language [7]. An alternative approach to provenance has been that of 'RDF Molecules' by Ding et al., which proposes a level of granularity (the 'molecule') that allows the original RDF statements to be re-constructed from disparate graphs [8]. Neither of these approaches have reached large-scale usage on the Semantic Web, much less standardization, and both approaches present a number of disadvantages. While the approach of RDF molecules allows one to reconstruct a graph, it does not allow the tracing of provenance of the graph if any of the information in the graph changes. Despite appeals to the fact that 'cool' URIs should not change,

the data hosted at URIs *will* change. For example, in my social-networking profile my organizational affiliation will likely change over time. How can we keep track of this? Proof-based systems around a variant of Datalog rules are too heavy-weight. Is it not odd that one can stick to standard RDF for describing data, but then move to a more expressive query and rule language to explain provenance? Also, the two most dominant Datalog-like Semantic Web languages, the upcoming W3C RIF language [2] and the N3 language [1], both present the syntactic problem that they themselves are not easily represented as RDF. While the W3C RIF language has been built so that it can deal with RDF data, the standard syntax of RIF is itself in idiosyncratic XML. N3 has its own syntax, that while close to the Turtle syntax for RDF, expands RDF in a number of ways that makes it semantically incompatible with RDF. Work on the Open Provenance Model for workflows is similar to RDF but seems to attempt to prematurely optimize an entire range of provenance operations without determining whether or not these operators can be derived from a few simple operators or are even necessary [12]. It seems a more advantageous route to some sort of provenance information would be to create a minimal vocabulary for provenance that would be expressible in standard RDF.

Recent work in provenance on relational databases has aimed precisely at creating such a minimal vocabulary, albeit for traditional relational data rather than RDF. In particular, the foundational work on provenance distinguishes between two kinds of provenance, the *where* provenance, which is the "locations in the source databases from which the data was extracted," and the *why* provenance, which is "the source data that had some influence on the existence of the data" [4]. Buneman et al. [4] present a Datalog-based model-theoretic semantics for calculating this kind of *where* and *why* provenance over queries. However, the traditional database community has been confronted with the same issues at the Semantic Web community with regards Datalog, as it would be better for most databases to keep the provenance in pure relational data. Therefore, current theoretical database work attempts to create more realistic models of provenance whose formal semantics do not rely on Datalog yet can still trace both the *where* and *why* provenance and can be implemented on top of run-of-the-mill SQL databases [3].

The most simple and accessible work from the database community is focused on providing a simple update language that tracks provenance, by focusing on three primary *provenance operators*: *insertion* (ins), *deletion* (del), and *copy* (copy). This is given by the grammar u, where for given fields in database q and p and a specific value v and a specific field a, $u ::= (ins\ a : v\ into\ p) \vee (del\ a\ from\ p) \vee (copy\ q\ into\ p)$ [3]. Buneman et al. considers this a 'cut-and-paste' model that can take into account the movement of data [3]. Such a model can then be easily implemented on top of normal databases by expanding the database so that sections of the database can store their provenance information, called the *provenance trace*. This trace can be queried, such that a user

should be able to query the ultimate 'source' of some data and its change history. Implementation-wise, every sector of rows and columns that has either been changed or copied from another database can have its provenance tracked by expanding the database with further rows and columns. So, whenever an insert, delete, or copy operation is committed, the corresponding provenance trace is tracked with identifiers for the source and target. A simple approach that records provenance with every interaction that changes the state of the database would lead to an explosion of database size, so current research is studying ways of optimizing provenance storage [3]. In an attempt to find a more solid semantic foundation for provenance, Cheney et al. [6] proposed a semantic characterization of provenance using functional dependency analysis, although they also proved that such a minimal dependency provenance is not computable, although dynamic and static techniques can approximate it. Another alternative for the semantics has been suggested by Green et al. [10], who used semi-rings to generalize algorithms over both *why*-provenance and relational algebras.

2 A Provenance Framework for the Semantic Web

It is our hypothesis that a simple vocabulary, composed of *insert*, *delete*, and *copy* operations as introduced by Buneman et al. provides a flexible format for provenance on the Semantic Web [3]. The current activity in the database community can then be used by the Semantic Web community in order to bootstrap a realistic implementation and scalable model of provenance. However, a few design issues are needed to be made in order to make the database approach compatible with the Semantic Web.

The first design choice is that the provenance operations be rephrased to operate over graphs rather than traditional relational data. Another question is whether or not separate *ins* and *copy* operators are needed by the Semantic Web. The *ins* operator in the context of databases (including XML databases) inserts a specific value, which may be a new value not in the graph or a replacement of a current value, at a determined location in the graph, while *copy* merely copies an entire graph. Thus, a *ins* function should be employed when one wishes to actually change or 'update' a given particular graph, as when changing the subject literal of a graph, while the *copy* function merely changes the name (often the host) URI of a graph or merges graphs, not necessarily changing any values. So, re-phrasing the provenance operator grammar u into RDF, given source graph G and target graph T as well as graph update a, $u ::= (ins\ a\ into\ T) \vee (del\ G\ from\ T) \vee (copy\ G\ into\ T)$. Traditional graph merge is then just when *ins* a into T without any deletion and where at least one node x is shared by at least one a and T.

The problem with phrasing provenance operators in RDF is that for the most part they deal with operations amongst entire graphs, and RDF lacks official support for identifying graphs. However, there is unofficial yet widely implemented

support in the form of *named graphs*, where each graph G is then identified with a *name URI* [5]. The idea of using named graphs for provenance is not new [5], but previously has been restricted to assertions that identify who is making a claim, not the provenance story championed by Buneman et al. [3]. If we assume the implementation of named graphs and the restriction of insertion values to (possibly typed) literal values, then we can phrase each of the provenance operations as RDF properties between the URIs of named graphs. The named graph approach solves the problem of *where* to store the provenance traces. Obviously, the provenance stores for each graph should be accessible, ideally using Linked Data principles, from the name URI of the named graph. This allows the name URI to ideally also serve as a SPARQL endpoint through which provenance queries can be done.

3 An Example of Provenance for Privacy and Trust

Most current efforts at privacy and trust within social networks make two assumptions. The first is that they assume that privacy can be handled via some sort of access control language, such as the unstandardized access language or an ontology like the Rei ontology developed by Kagal [11]. Assuming that policy languages can be phrased in terms of access control makes a certain amount of sense. Yet at minimum, this viewpoint is predicated upon certain behavior being either explicitly allowed or not, with the main behavior likely being the copying (equivalent to 'viewing,' as with a view one can doubtless copy), insertion, and deletion operations of the provenance. So, any access control language should operate over at least the three provenance operators. Second, Semantic Web research like Goldbeck's Trust Ontology makes the radically simplistic assumption that trust can be quantified as integer-valued rating between 1 to 10 (or some other arbitrary numbers, such as real-valued rating between 0 and 1) [9]. Where do such ratings come from? Obviously, they may come from the often unreliable subjective impression of the users. It is far better to calculate 'trust' from actual data sources and its changes. Therefore, it is of utmost concern to track the provenance of the data, and the use of provenance traces with named graphs allow such trust ratings to be calculated in a principled and algorithmic manner, and then privacy constraints built on top of such trust ratings.

For a quick example, we will use the infamous real-world 'dump your pen friend' scenario.[1] A young student has her picture taken and uploaded by a friend to Flickr with a Creative Commons license that allows commercial use. A major company finds the photo on Flickr and proceeds to use it in an advertising campaign across Australia (where the student lives), with an unflattering slogan beneath the photo that says "Dump Your Pen Friend." The young student feels emotionally damaged and sues the company in court. The problem at hand can be confronted one of two ways. The first is to assume that the owner of the photo

[1] *http://ipandentertainmentlaw.wordpress.com/2007/10/15/update-dumb-your-pen-friend/*

perhaps did not fully understand what situation they were putting their friend in by releasing their photo under the Creative Commons License, in which case all that is needed is to offer a more full explanation of the license. However, it is more likely to assume that the original owner of the photo understood Creative Commons, but did not notify their friend that an unflattering picture of them was being uploaded that could be used for-profit by a company, and the company did not alert the young student. However, in order to both determine who was in the picture and the chain of events of copying the photo that led to the incident, what precisely is needed is *provenance*.

One can see how a provenance framework could have led to a solution to this scenario. When the photo was originally taken, the provenance information about the date the picture was taken and its owner could be taken, phrased as FOAF RDF statements involving the URI of the picture, using *ex* as the *http://www.example.org/* namespace, *foaf:* is *http://xmlns.com/foaf/spec/*, *prov:* as a yet undetermined provenance namespace, and all examples given using the Turtle syntax (assuming the student whose picture was taken is 'Jane Doe' and the taker of the picture was 'John Doe'). So the graph *ex:photo.jpg rdf:type foaf:Image, foaf:maker ex:JohnDoe* could be given a name URI *ex:thephoto#it*. When the picture was uploaded to a popular social-networking site, the student could have seen themselves in the photo and so 'tagged' themselves in the photo, and thus inserted a new value using *prov:ins*, with the new value being the triple *ex:newtriple* which contains *ex:photo.jpg foaf:depicts ex:JaneDoe*. This provenance information is recorded using *ex:/photo#it prov:ins ex:newtriple*. After the photo was uploaded, the company then copied the photo, and this is stored in the provenance trace of the named graph by using the *prov:copy* operator. Assuming the company's URI to be *http://thecompany.info*, this could be recorded as *ex:photo#it prov:copy http://thecompany.info/copyOf/photo#it*. Relevant temporal information could also be added to the named graph records in the provenance trace. Therefore, the provenance trace of the photo could be followed, so that before the photo was used in a nation-wide advertising campaign, the company and student both could determine who was in the photo.

4 Conclusion and Future Work

We have argued so far in this paper that a simple model of provenance, based around the central three provenance operators and the storage of provenance traces, can be used on the Semantic Web by merging it with an approach based on named graphs. This is at best a sketch towards the completion of a provenance-aware Semantic Web. A small vocabulary of provenance operators need to be standardized and given a namespace URI. Second, the interaction of provenance operators and traces with RDF and named graphs needs to be given a formal semantics and implemented. This step may not be as difficult as it seems, as simple operations such as 'copy' or *where* provenance are already implemented by many as the default use of named graphs, and the deletion and

copy operators are already implemented in the form of the proposed SPARQL Update language [14]. There has already been work on the formal semantics for provenance operators and traces within the nested relational calculus [6] as well as giving RDF provenance over named graphs a formal semantics [13]. So, a practical RDF vocabulary for provenance with a formal semantics should be possible to deploy and standardize.

References

1. T. Berners-Lee, D. Connolly, L. Kagal, Y. Scharf, and J. Hendler. N3 Logic: A logic for the Web. *Journal of Theory and Practice of Logic Programming*, pages 249–269, 2007.
2. H. Boley and M. Kifer. RIF Basic Logic Dialect. Working draft, W3C, 2008. http://www.w3.org/TR/rif-bld/ (Last accessed Aug. 8th 2008.
3. P. Buneman, A. Chapman, and J. Cheney. Provenance management in curated databases. In *SIGMOD '06: Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pages 539–550, New York, NY, USA, 2006. ACM Press.
4. P. Buneman, S. Khanna, and W. chiew Tan. Why and where: A characterization of data provenance. In *In ICDT*, pages 316–330. Springer, 2001.
5. J. Carroll, C. Bizer, P. Hayes, and P. Stickler. Named graphs. *Journal of Web Semantics*, 4(3):247–267, 2005.
6. J. Cheney, A. Ahmed, and U. A. Acar. Provenance as dependency analysis. *CoRR*, abs/0708.2173, 2007.
7. P. daSilva, D. McGuinness, and R. Fikes. A proof markup language for semantic web services. *Information Systems*, 31:381–395, 2006.
8. L. Ding, T. Finin, Y. Peng, P. P. da Silva, and D. L. McGuinness. Tracking RDF graph provenance using rdf molecules. In *Proc. of the 4th International Semantic Web Conference (Poster)*, 2005.
9. J. Goldbeck. Trust ontology, 2006. http://trust.mindswap.org/ont/trust.owl (Last Accessed March 5th 2009.
10. T. J. Green, G. Karvounarakis, and V. Tannen. Provenance semirings. In *PODS '07: Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 31–40, New York, NY, USA, 2007. ACM.
11. L. Kagal, T. Berners-Lee, D. Connolly, and D. Weitzner. Using Semantic Web Technologies for Open Policy Management on the Web. In *Proceedings of the National Conference of the Association for Artificial Intelligence*, 2006.
12. L. Moreau, J. Freire, J. Futrelle, R. McGrath, J. Myers, and P. Paulson. The open provenance model, December 2007.
13. P. Pediaditis, G. Flouris, I. Fundulaki, and V. Christophides. On explicit provenance management in rdf/s graphs. In *Workshop on the Theory and Practice of Provenance*, 2009.
14. A. Seaborne, G. Manjunath, C. Bizer, J. Breslin, S. Das, I. Davis, S. Harris, K. Idehen, O. Corby, K. Kjernsmo, and B. Nowack. SPARQL update: A language for updating RDF graphs. Member submission, W3C, 2008. http://www.w3.org/Submission/2008/SUBM-SPARQL-Update-20080715/ (Last accessed March 10th 2009).