Extending Policy Languages to the Semantic Web

E. Damiani, S. De Capitani di Vimercati, C. Fugazza, and P. Samarati

DTI - Università di Milano
26013 Crema - Italy
{damiani,decapita,samarati}@dti.unimi.it, fugazza@dsi.unimi.it

Abstract. In the semantic web environment it is important to be able to specify access control requirements about subjects accessing the information and about resources to be accessed in terms of the rich ontology-based metadata describing them. In this paper, we outline how current standard policy languages such as XACML can be extended to address this issue. Then, we describe a reference architecture for enforcing our semantics-aware policies.

1 Introduction

Current web-based applications describe resources (including users) and services via a number of XML-based standard protocols [10]. Among those, XML-based standard definitions for policies and credentials have recently received an increased interest [5, 13]. *Semantic web technologies* are changing this picture, using advanced knowledge representation techniques for enriching e-business environments. The semantic web is built around the notion of representing shared knowledge via standard *ontologies*, that are used by intelligent agents to understand the nature of the information they are processing [6]. In an interoperable e-business architecture based on the semantic web vision, *ontology-based domain models* are therefore used as controlled vocabularies for resources description, allowing users to obtain the right resources at the right time [3]. While research on developing standards and tools that ultimately will lead to the existence of the semantic web is increasing [16], security impacts of these new technologies have not been addressed sufficiently. On the semantic web, writing access control policies where subjects and objects are pointed at via data identifiers or simple predicates is not enough. Rather, it is important to be able to specify access control requirements about subjects and resources in terms of the rich metadata describing them.

Some researchers have recently investigated security within the semantic web for the purpose of either expressing security policies or protecting semantically rich data. As an example of the two, [4, 7] develop security ontologies that allow parties to share a vocabulary to exchange security-related information using a common language; while [9, 12] present policy languages to specify access restrictions over concepts defined in ontologies. Another line of work merging security and semantic web concepts is presented in [14] as an approach for identifying

Web inference channels due to ontology-based inference attacks. There, an ontology is used to detect tags appearing in different XML documents that are *ontologically equivalent* (i.e., can be abstracted to the same concept in the ontology), but which have contradictory security classifications.

Although these approaches represent a first step toward the definition of a semantics-aware access control process, they do not completely exploit the power of the semantic web. Neither semantic web-based proposals nor emerging stateful attribute-based security languages (e.g., XACML) allow support of access restrictions on resources according to complex semantics-aware assertions.

In this paper we bring forward the idea of exploiting the semantic web to extend current policy languages to allow the definition of access control rules based on generic assertions defined over concepts in the ontologies that control metadata content and provide abstract subject domain concepts, respectively. These rules are then enforced on resources annotated with metadata regulated by the same ontologies. Our proposal allows for specifying access control requirements about: *i)* subjects accessing the information and *ii)* resources to be accessed in terms of rich ontology-based metadata associated with them. The result is a powerful policy language exploiting the high expressive power of ontology-based models.

The contribution of this paper can be summarized as follows. First, we make the point for the need of more expressive policy languages and show how semantic web metadata formats can be exploited to this end. Second, we show how current standard security languages can be extended to include semantic-aware specifications. Third, we illustrate our architectural solution, including a semantic-aware authentication tool, called OntoPassport. Our proposal extends the eXtensible Access Control Markup Language (XACML) [5] by adding the capability to designate subjects and objects via generic RDF statements [15]. Our extensions to XACML introduce the following capabilities.

- *Semantic-aware subject description.* Semantic web applications need to easily operate on subject descriptions to determine whether a policy rule is applicable to a user.
- *Canonical metadata syntax.* The high expressive power of semantic web metadata allows for using different syntax to carry the same semantics. While no constraints can be posed apriori on the content of resources' descriptors, a standard syntax could be adopted for metadata used to describe subjects and objects within access control policies. Also, a standard syntax could be used for subjects' descriptions.

2 Basic concepts

In this section, we briefly overview the main standards on which our proposal is based, namely, XACML, SAML (the XML standard for encapsulating access requests) and RDF, highlighting the features relevant to our research.

XACML

XACML is the result of a recent OASIS standardization effort proposing an XML-based language to express and interchange access control policies. XACML is designed to express authorization policies in XML against objects that are themselves identified in XML. The language can represent the functionalities of most policy representation mechanisms. An XACML policy consists of a set of rules whose main components are: a target, an effect, and a condition.[1] The target defines the set of resources, subjects, and actions to which the rule is intended to apply. The effect of the rule can be `permit` or `deny`. The condition represents a boolean expression that may further refine the applicability of the rule. A request consists of attributes associated with the requesting subject, the resource involved in the request, the action being performed and the environment. A response contains one of four decisions: `permit`, `deny`, `not applicable` (when no applicable policies or rules could be found), or `indeterminate` (when some errors occurred during the access control process). A request, a policy, and the corresponding response form the `XACML Context`.

The subjects and resources to which a rule applies are defined by using a set of pre-defined functions (e.g., equality, set comparison, arithmetic) and datatypes (e.g., string, boolean, integer). To illustrate, consider the following portion of a rule.

```
......
<Subject> <!-- match role subject attribute -->
  <SubjectMatch
    MatchId= "urn:oasis:names:tc:xacml:1.0:function:string-equal"> <!-- function -->
    <AttributeValue DataType= "http://www.w3.org/2001/XMLSchema#string">
      physician
    </AttributeValue>
    <SubjectAttributeDesignator AttributeId=
      "urn:oasis:names:tc:xacml:1.0:example:attribute:role"
      DataType= "http://www.w3.org/2001/XMLSchema#string" />
  </SubjectMatch>
</Subject>
......
```

This rule matches requests where the subject has an attribute `role` whose value is `physician`. While these functions and datatypes can be used to define many access control policies, XACML also specifies an extension mechanism for defining additional datatypes and functions. Our proposal makes use of such a mechanism to accommodate semantically rich constraints.

SAML

On corporate networks as well as on the global Net, access to services by single-sign-on authentication is becoming a widespread way to authenticate users.

[1] We keep at a simplified level the description of the language and refer the reader to the OASIS proposal [5] for the complete specification.

Single-sign-on authentication relying on digital certificates is currently the most popular choice for e-business applications. In this area, the most successful standard is SAML [13], an authentication protocol handling authentication information across transactions between parties. SAML uses tagged sets of user attributes to represent subject-related information encapsulated inside service requests. One might wonder whether the semantics of user credentials could be represented directly by the standard XML schemata describing SAML. Unfortunately, standard XML schema definitions need to cover a wide repertoire of possible user attributes. For this reason, optional elements are widely used, thus decreasing the expressiveness of the schema as a descriptor of single instances. The goal of authentication standards is to carry information according to a protocol, rather than describing a domain; also, a considerable number of SAML tags have a structural function and do not describe specific subjects.

RDF

RDF is a model to express generic assertions (RDF statements) associated with resources. An RDF statement is a triple of the form (*subject*, *predicate*, *object*), where subject[2] is the resource being described, predicate is a property associated with the resource, and object is the value of the property. For instance, statement `"index.html has been created by Lucy"` can be represented by the triple (index.html, created_by, Lucy). Due to the fact that the subject of a statement can be any resource, it is possible to make statements about statements in RDF. This technique is called *reification*. To clarify this important concept, consider the following statement:

“`Video http://www.acme.com/medical.avi shows Dr. Harry Morris assisting patient Carl Smith`”

To state this in RDF one must define several RDF statements:

```
(Harry Morris, is-a, physician);
(Carl Smith, is-a, patient);
(Harry Morris, assists, Carl Smith);
(http://www.acme.com/medical.avi, is-a, Video);
(http://www.acme.com/medical.avi, shows, ‘‘Harry Morris, assists, Carl
Smith’’).
```

Reification is represented by splitting a statement into three different parts (subject, predicate, and object) and asserting these parts. By reification, the previous sentences can be expressed as the following set of triples.[3]

```
(http://www.acme.com/medical.avi, type, Video)
(Harry Morris, type, Physician)
(Carl Smith, type, Patient)
(assists, type, relation)
(shows, type, relation)
```

[2] This is not to be confused with the use of subject in the authorization model.

[3] For the sake of clarity, we shall use an informal syntax where capital letters represent URIs and drop the use of XML namespace prefixes.

```
(A, type, statement)
(A, subject, Harry Morris)
(A, predicate, assists)
(A, object, Carl Smith)
(B, type, statement)
(B, subject, http://www.acme.com/medical.avi)
(B, predicate, shows)
(B, object, A).
```

Reification is used often and, in our approach, a uniform RDF based on reification simplifies the evaluation procedure.

3 Towards semantic-aware access control language

In this section we show how current XML-based standards can be extended to seamlessly incorporate metadata-based designations of subjects and objects.

3.1 Including assertion-based metadata in XACML

The design of a policy evaluation and enforcement engine exploiting semantic web metadata needs to be based on a sound model and language for expressing authorizations in term of metadata. To this purpose, we chose to exploit the *extensibility points* already built in the XACML language rather than redesigning a policy language from scratch. Our extension points can be summarized as follows.

- Extend the `XACML Context` to include metadata associated with both subjects and resources.
- Extend the `AttributeValue` XACML element (used in XACML to qualify both subjects and objects) capability of specifying auxiliary namespaces.[4] Auxiliary namespaces to be added are at least two: the `rdf:` one, allowing for using RDF assertions as values for the XACML `AttributeValue` element and another one (in our example, `md:` and `ms:`) enabling using properties and class names from a user ontology within those assertions.
- Extend the `MatchID` attribute by introducing a new function, called `metadataQuery`, expressing the processing needed for policy enforcement.

Although our proposed extensions to XACML rely on standard RDF syntax, some precautions should be taken to keep enforcement possible; namely, we prescribe that attribute values written in RDF use a RDF reification technique.

[4] Such additional attribute values are optional and do not disrupt parsability of standard XACML policies using our extended schema.

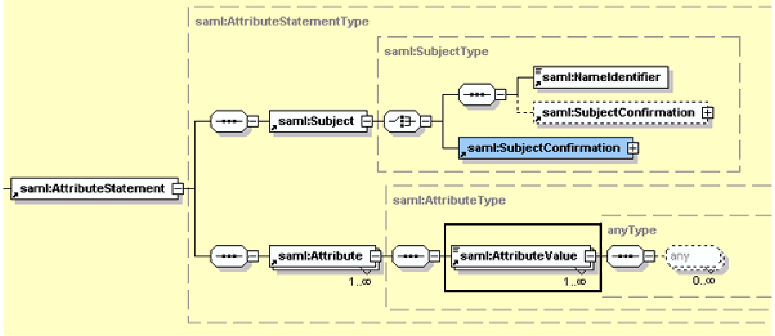

Fig. 1. The Schema for a SAML assertion

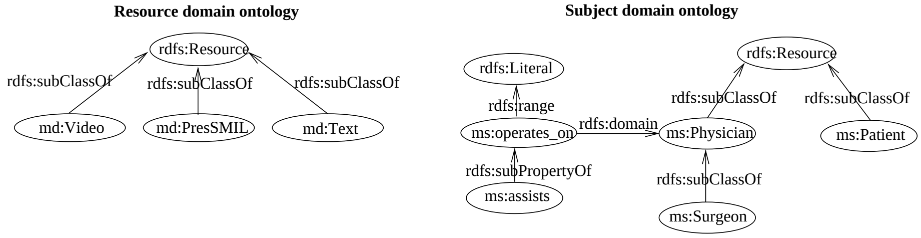

Fig. 2. An example of resource and subject domain ontology

3.2 Incapsulating semantics-aware credentials in SAML

Figure 1 shows the portion of SAML-XML Schema specifying the structure of an authentication assertion, the boxed element shows our proposed extension point. Basically, the SAML schema defines a subject identification and associate it with a set of attributes. The attribute definition is extremely open, leaving it to application-specific XML schemata to specify the actual set of attributes identifying the user. We extend the attributes allowed for the `AttributeValue` element to enable content including RDF assertions using suitable ontology concepts as predicate names.

In the simplest case, the subject metadata can assert that the user holding the certificates belongs to a certain type such as `(thisRequestUser, type, Physician)`, or more complex ones such as

```
(thisRequestUser, type, Person)
(thisRequestUser, buys , "Resource")
(Resource, type, MovieDVD)
(Resource, title, "Lord of the Rings")
```

However, once again we use a canonical reified syntax.

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
  xmlns:md="http://ourdomain.it/MD/Schema/md-syntax#"
  xmlns:ms="http://ourdomain.it/MS/Schema/ms-syntax#">
  <rdf:Description
    rdf:about="http://ourdomain.it/MD/Video/video010234.avi">
    <rdf:type rdf:resource="http://ourdomain.it/MD/Schema/md-syntax#Video" />
    <md:title>Treatment of Diseases</md:title>
    <md:duration>1054067</md:duration>
    <md:format>avi</md:format>
    <md:shows_how rdf:nodeID="content"/>
  </rdf:Description>
  <rdf:Description rdf:nodeID="content">
    <ms:surgeon>Sam</ms:surgeon>
    <ms:operates_on>Patient</ms:operates_on>
  </rdf:Description>
</rdf:RDF>
```

Fig. 3. An example of RDF metadata associated with a video presentation

3.3 Using the extended language

To illustrate our examples of semantics-aware access control policies, we shall consider a *medical digital library* (MDL) that includes different kinds of multimedia data: free text, images, video, and audio. Each multimedia data item is complemented with descriptive metadata in the form of RDF descriptors. RDF descriptors could contain any assertion about the data items that can be written using the ontology vocabulary. However, in some controlled environments it might be possible to adopt the reification-based syntax greatly simplifying the evaluation procedure. In the following, we shall assume that the reified format of RDF statements is used. Note that however conversion tools are available capable to translate a variety of RDF syntax into the reified ones.

To express the statements in our descriptors, we use three vocabularies. The first vocabulary contains standard terms such as `predicate`, `statement`, `subject`, `object` and standard relations such as `is-a`. This vocabulary is the RDFS base namespace (whose formal definitions can be found in [15]) containing the elements `rdf:statement, rdf:subject, rdf:predicate, rdf:object, rdf:type` that have a self-explanatory semantics. The second vocabulary, called *resource domain ontology*, contains domain-specific terms that are used to describe the resource content such as `Video` and `shows_how`. The third vocabulary, called *subject domain ontology*, contains terms that are used to make assertions on subjects such as `Physician`, `Patient`, `assists`. Figure 2 illustrates the resource domain ontology and subject domain ontology where, for the sake of simplicity, we only report the main concepts that will be used to show the expressive power of our proposal. Note that generally speaking no assumption can be made about the format of RDF descriptors associated with resources. Figure 3 illustrates an example of RDF descriptor where, in addition to the classical `rdf:` namespace, we use namespace `md:` for describing multimedia data and namespace `ms:` for describing medical staff. The RDF descriptor, associated with a video (`video010234.avi`), comprises of two sections: the first one expresses the document type according to a domain ontology and to some additional simple metadata (e.g., format of the document, title); the second one contains more complex metadata expressing the fact that "Video `video010234.avi` shows how surgeon Sam operates on a patient". Note that while the simple metadata in the first section could be expressed by a usual attribute-value pair, this is not the case for advanced metadata, to which our approach applies. Consider now the following protection requirement:

> Physicians of the Trauma Therapy Center department are allowed to see video presentations that show physicians assisting patients

This requirement is composed of two assertions stating, respectively, *1)* who can access the resource (Physicians of the Trauma Therapy Center department) and *2)* the kind of resources involved (Video presentations that show physicians assisting patients). Such assertions are used to define the target of the XACML rule as illustrated in Figure 4. Consider now a request to see video video010234.avi submitted by a user who presents to our system subject metadata stating that the requester is Sam, a surgeon of the Trauma Therapy Center Department (see RDF description in Figure 5). Intuitively, by exploiting the hierarchical organization of the concepts defined in the domain ontologies, the evaluation of this access request should return a permit decision because both Sam and the video involved in the request satisfy the subject and resource conditions specified in the rule, respectively. This is the result of the two following subsumptions:

- Surgeon is a sub-class of Physician
- assists is a sub-property of operates_on.

We will see in more details the policy evaluation process in the next Section.

4 Policy evaluation

When a policy involving metadata needs to be evaluated, the subject context already contains the RDF description of the requester, taken from the SAML request. Our policy evaluation engine works as follows.

1. The semantic assertions about the requester that are included in the subject field of our policy rules and the metadata about the requester in the access request are compared to identify the policy rules that apply to the requester.
2. The semantic assertions that are included in the resource context of applicable policy rules are used to query the descriptive metadata of the requested resource, to verify whether the requested resource satisfies the rules selected in the previous step.

Both these selection steps involve RDF queries, where the assertions in the policy rules are used to query metadata associated with the requester and the involved resource.[5] A suitable query language is DQL, a logic-based query language

[5] Such querying can be tackled by means of two different techniques: *reasoning* based on metadata and *database-like* querying. The former approach considers RDF metadata as a knowledge base that can be translated into logic programming clauses and applies reasoning techniques to them.

```
<?xml version="1.0" encoding="UTF-8"?>
<Rule
  xmlns="urn:oasis:names:tc:xacml:1.0:policy"
  xmlns:xsi= http://www.w3.org/2001/XMLSchema-instance
  xmlns:ctx="urn:oasis:names:tc:xacml:1.0:context"
  xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
  xmlns:md="http://ourdomain.it/MD/Schema/md-syntax"
  xmlns:ms="http://ourdomain.it/MS/Schema/ms-syntax"
  RuleId="urn:oasis:names:tc:xacml:examples:ruleid:1"
  Effect="Permit">
  <Target>
    <Subjects>
      <Subject>
        <SubjectMatch
          MatchId= "urn:ourdomain:function:metadataQuery">
          <AttributeValue
            DataType="http://">
            <rdf:Statement rdf:about="thisRequestUser" >
               <rdf:subject rdf:resource="http://ourdomain.it/MS/Schema/ms-syntax#Physician" />
               <rdf:predicate rdf:resource="http://ourdomain.it/MS/Schema/ms-syntax#belongs"/>
               <rdf:object rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                     Trauma Therapy Center
               </rdf:object>
            </rdf:Statement>
          </AttributeValue>
        <SubjectAttributeDesignator
          AttributeId="urn:ourdomain:attribute:metatag"
          DataType="http://www.w3.org/2001/XMLSchema#string"/>
        </SubjectMatch>
      </Subject>
    </Subjects>
    <Resources>
      <Resource>
        <ResourceMatch
          MatchId="urn:ourdomain:function:metadataQuery">
          <AttributeValue
            DataType="http://">
            <rdf:Statement rdf:about="thisRequestUrl">
               <rdf:subject rdf:resource="http://ourdomain.it/MS/Schema/md-syntax#Video"/>
               <rdf:predicate rdf:resource="http://ourdomain.it/MD/Schema/md-syntax#shows_how"/>
               <rdf:object rdf:nodeID="content"/>
            </rdf:Statement>
            <rdf:Statement rdf:nodeID="content">
               <rdf:subject rdf:resource="http://ourdomain.it/MS/Schema/ms-syntax#Physician"/>
               <rdf:predicate rdf:resource="http://ourdomain.it/MS/Schema/ms-syntax#assists"/>
               <rdf:object rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                     Patient
               </rdf:object>
            </rdf:Statement>
          </AttributeValue>
        <ResourceAttributeDesignator
          AttributeId="urn:ourdomain:attribute:metatag"
          DataType="http://www.w3.org/2001/XMLSchema#string"/>
        </ResourceMatch>
      </Resource>
    </Resources>
    <Actions>
      <Action>
        <AnyAction />
      </Action>
    </Actions>
  </Target>
</Rule>
```

Fig. 4. An example of access control policy in extended XACML

for the semantic web proposed in [2]. Here, however, we follow an SQL-like or an XQuery approach, assuming that RDF metadata about resources are stored as a relational or an XML database.

First, let us examine the rule selection step. Suppose a request comes in whose encapsulated metadata are:

```
(A, type, statement)
(A, subject, thisRequestUser)
(A, predicate, type)
(A, object, Physician)
```

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
  xmlns:ms="http://ourdomain.it/MS/Schema/ms-syntax#">
  <rdf:Statement
    rdf:about="http://ourdomain.it/MS/medstaff/11234">
    <rdf:subject rdf:resource="http://ourdomain.it/MS/Schema/ms-syntax#Surgeon" />
    <rdf:predicate rdf:resource="http://ourdomain.it/MS/Schema/ms-syntax#belongs"/>
    <rdf:object rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
      Trauma Therapy Center
    </rdf:object>
  </rdf:Statement>
  <rdf:Statement
    rdf:about="http://ourdomain.it/MS/medstaff/11234">
    <rdf:subject rdf:datatype="http://www.w3.org/2001/XMLSchema#rfc822Name" >
      http://ourdomain.it/MS/Schema/ms-syntax#Surgeon"
    </rdf:subject>
    <rdf:predicate rdf:resource="http://ourdomain.it/MS/Schema/ms-syntax#name"/>
    <rdf:object rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
      Sam
    </rdf:object>
  </rdf:Statement>
</rdf:RDF>
```

Fig. 5. An example of RDF description associated with a requester

Then all XACML rules R whose subject metadata include `(?, subject, Surgeon)` will be selected.

Let us assume that the resource metadata mentioned in the context of the policy rule R is the following:

```
(?, type, Statement)
(?, subject, Physician)
(?, predicate, assists)
(?, object, Patient)
```

These metadata can now be used to build a query on the resource descriptors, to identify the objects to which the rule applies. For instance, the policy will apply to the video presentation with the metadata shown in Figure 3.

The reified statement contained in the policy is used to construct the query which is submitted to the set of resource descriptors. Therefore, to evaluate the feasibility of our approach, the complexity of RDF query answering must be taken into account. RDF query evaluation process is composed of three main phases [8]: matches computation, minimization of queries, and redundancy elimination in answers. Here we shall only discuss the first contribution since some theoretical results are available from RDF querying research. The other two phases will need to be addressed with ad hoc solution for policy evaluation.[6] In [8] the authors used the simpler problem of testing emptiness of the query answer as an approximation of the complexity of computing the matches. Distinguishing between *query complexity*, considering the evaluation time as a function of query size for a given database, and *data complexity*, considering the evaluation time of a given query as a function of database size for a fixed query, they find out that evaluation is NP-complete for the query complexity and polynomial for the database complexity version. Furthermore, the size of the set of answers of a query q issued against a database D is bounded by $|D||q|$, where $|D|$

[6] Since query evaluation is often exponential in query size, static optimization of queries is an important goal.

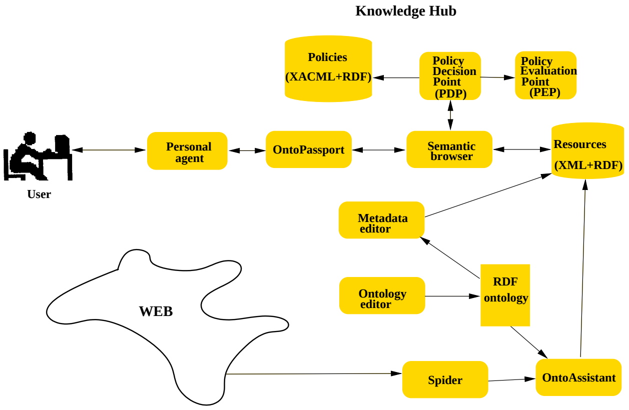

Fig. 6. Our reference semantic web architecture

is the size of the database (number of triples) and $|q|$ is the number of symbols in the query.

5 Our architecture: the OntoPassport

To present our architecture we will focus on our semantic web system, called *Knowledge Hub* (KH) [1], illustrated in Figure 6. In the KH, network resources are first downloaded in their native XML/XHTML format[7], tidying them up if necessary. Then, a special-purpose tool, called *OntoAssistant*, is used to produce RDF descriptions of their content. Such descriptions are written exploiting the standard vocabulary provided by a business ontology, written in the RDFS standard language. While more structured ontology description languages such as OWL [17] are now available, the KH setting is general enough for our current purposes. Users connect to the KH system via a Semantic Navigator that allows them to navigate/query metadata as well as the data themselves. Thanks to the body of knowledge comprising the ontologies and the metadata, the Semantic Navigator can act as a logic program, performing more sophisticated reasoning than a conventional query engine. Also, the Semantic Navigator can use the available metadata about resources and users themselves to customize their navigation experience and data presentation. To present a semantic web application like our KH with a semantics-aware description of a user, we can encapsulate ontology-based metadata about the user within a SAML request. Our

[7] Of course, it would be perfectly feasible to associate metadata with external resources without downloading them; however, this would bring up the problem of keeping data and metadata consistent, which is outside of the scope of this paper.

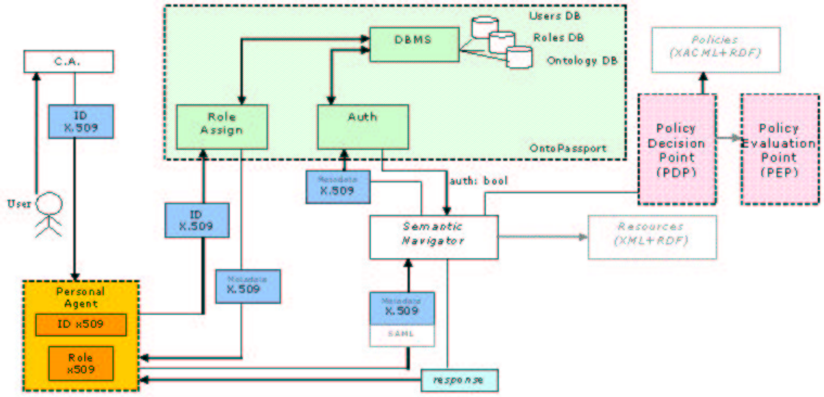

Fig. 7. The architecture of a security solution including the OntoPassport component

OntoPassport tool was designed to handle this approach by encoding metadata in a X.509 certificate that can be carried by SAML.[8]

Figure 7 shows a closer view of our evaluation architecture. Figure 8 shows the WSDL interface definition for the OntoPassport implemented as a web service. The input datatypes correspond to the XML encoding of a generic certificate, while the return datatype is designed to cointain a set of RDF assertions that qualify the user with respect to a suitable role ontology. We could substitute the username-password authentication scheme with a traditional certificate. In this case the OntoPassport acts as a "translator" of an authentication certificate into a digital identity according to an ontology.

The OntoPassport also includes in the certificate the name, URL and version of the RDFS Schema specifying the ontology it used to build the subject metadata. This makes the type hierarchy used for subjects (and, more generally, the entire metadata vocabulary used to talk about them) easy to share and maintain across an organization. We are currently working on the implementation of a PEP component including a fully featured RDF query engine.

6 Conclusions

Traditional access control models and languages result limiting for emerging Web applications. Although recent advancements allow the specifications of access control rules with reference to generic attributes/properties of the requestor

[8] Alternatively, our semantics-aware metadata can be saved in a secure cookie [11] on the user's machine.

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="MyWebService"
  targetNamespace="http://MyServer/OntoPassport.wsdl"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:tns="http://MyServer/OntoPassport.wsdl"
  xmlns:ns1="http://MyServer/MyWebService.xsd">
  <types>
    <schema targetNamespace="http://MyServer/MyWebService.xsd"
      xmlns="http://www.w3.org/2001/XMLSchema"
      xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/">
      <complexType name="OntoPassportToken">
        <all>
          <element name="name" type="string"/>
          <element name="idCode" type="string"/>
          <element name="surname" type="string"/>
          <element name="mail" type="string"/>
          <element name="privilege" type="string"/>
          <element name="ttl" type="int"/>
          <element name="timeStamp" type="int"/>
        </all>
      </complexType>
    </schema>
  </types>
  <message name="tokenRequestRequest">
    <part name="usr" type="xsd:string"/>
    <part name="pwd" type="xsd:string"/>
  </message>
  <message name="tokenRequestResponse">
    <part name="return" type="ns1:OntoPassportToken"/>
  </message>
  <portType name="OntoPassportType">
    <operation name="tokenRequest">
      <input name="tokenRequestRequest" message="tns:tokenRequestRequest"/>
      <output name="tokenRequestResponse" message="tns:tokenRequestResponse"/>
    </operation>
  </portType>
  <binding name="OntoPassportBinding" type="tns:OntoPassportPortType">
    <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="tokenRequest">
      <soap:operation soapAction="" style="rpc"/>
      <input name="tokenRequestRequest">
        <soap:body use="encoded" namespace="MyWebService"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
      </input>
      <output name="tokenRequestResponse">
        <soap:body use="encoded" namespace="MyWebService"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
      </output>
    </operation>
  </binding>
  <service name="MyWebService">
    <port name="OntoPassportPort" binding="tns:OntoPassportBinding">
      <soap:address location="http://OntoServices/MyWebService"/>
    </port>
  </service>
</definitions>
```

Fig. 8. The OntoPassport WSDL interface

and the resources, they do not fully exploit the semantic power and reasoning capabilities of emerging web applications. We have presented a semantics-aware approach aimed at controlling access to resources on the basis of complex assertions about subjects seeking access as well as about resources, stated by means of semantic web metadata standards. We have also shown how this expressive power can be easily accommodated by proper extensions of available XML-based policy languages, like XACML. While several aspects (including efficient techniques for performing enforcement) are still to be investigated, our proposal provides a clear problem statement and a first step toward its solution.

7 Acknowledgments

This work was supported in part by the European Union within the PRIME Project in the FP6/IST Programme under contract IST-2002-507591 and by the Italian MIUR within the KIWI and MAPS projects.

References

1. A. Corallo, E. Damiani, and G. Elia. A knowledge management system enabling regional innovation. In *Proc. of the International Conference on Knowledge-Based Intelligent Information Engineering Systems & Allied Technologies (KES 2002)*, Crema, Italy, September 2002.
2. DAML query language (DQL), April 2003. http://www.daml.org/2003/04/dql/.
3. J. Davies, D. Fensel, and F. van Harmelen. *Towards the Semantic Web: Ontology-Driven Knowledge Management*. John Wiley & Sons, Ltd, 2002.
4. G. Denker, L. Kagal, T. Finin, M. Paolucci, and K. Sycara. Security for DAML web services: Annotation and matchmaking. In *Proc. of the 2nd International Semantic Web Conference (ISWC2003)*, Sanibel Island, Florida, USA, October 2003.
5. eXtensible Access Control Markup Language. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml.
6. D. Fensel. *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*. Springer-Verlag, 2003.
7. T. Finin and A. Joshi. Agents, trust, and information access on the semantic web. *ACM SIGMOD*, 31(4):30–35, December 2002.
8. C. Gutierrez, C. Hurtado, and A. Mendelzon. Formal aspects of querying RDF databases. In *Proc. of First International Workshop on Semantic Web and Databases*, Berlin, Germany, September 2003.
9. L. Kagal, T. Finin, and A. Joshi. A policy based approach to security for the semantic web. In *Proc. of the Second International Semantic Web Conference (ISWC2003)*, Sanibel Island FL, October 2003.
10. R. Khosla, E. Damiani, and W. Grosky. *Human-centered E-business*. Kluwer Academic Publisher, 2003.
11. J.S. Park and R.S. Sandhu. Secure cookies on the web. *IEEE Internet Computing*, 4(4):36–44, 2000.
12. L. Qin and V. Atluri. Concept-level access control for the semantic web. In *Proc. of the ACM Workshop on XML Security 2003*, Fairfax, VA, PA, October 2003.
13. Security assertion markup language (SAML) v1.0. http://www.oasis-open.org/committees/download.php/3400/oasis-sstc-saml-1.1-pdf-xsd.zip.
14. A. Stoica and C. Farkas. Ontology guided security engine. *Journal of Intelligent Information Systems*, 2004.
15. World Wide Web. *RDF Vocabulary Description Language 1.0: RDF Schema*, December 2003. http://www.w3.org/TR/rdf-schema/.
16. World Wide Web Consourtium. *Semantic Web*. http://www.w3.org/2001/sw/.
17. World Wide Web Consourtium. *OWL Web Ontology Language – Overview*, December 2003. http://www.w3.org/TR/owl-features/.