Supporting Attribute-based Access Control with Ontologies

Torsten Priebe
Capgemini Consulting Österreich AG
Lassallestraße 9b, A-1020 Vienna, Austria
torsten.priebe@capgemini.com

Wolfgang Dobmeier, Nora Kamprath
Department of Information Systems,
University of Regensburg, D-93040 Regensburg, Germany
wolfgang.dobmeier@wiwi.uni-regensburg.de

Abstract

In highly open systems like the Internet, attribute-based access control (ABAC) has proven its appropriateness. The specification and maintenance of ABAC policies however has turned out to be complex and error-prone, especially if heterogeneous attribute schemes are involved. Here, the arising Semantic Web can contribute to a solution. This paper presents an approach based on an extension of the established XACML standard. It simplifies the policies by providing an ontology-based attribute management facility.

1. Introduction

Due to the growing use of the Internet, more and more critical processes are run over the Web. Examples are e-government or e-commerce applications, or solutions within large organizations like enterprise portals. These have to be protected from unauthorized access in an adequate manner, e.g., commercial information services should be accessible only by paying customers. So far, models like role-based access control (RBAC) [4] were used in such environments. However, these approaches have deficiencies in large open systems, where the number of potential users is very high and most users will not be known beforehand. As the RBAC model is not flexible enough to deal with these requirements, the use of attribute-based technologies (ABAC) [7, 8] has been introduced, e.g., within the XACML standard [6]. However, the higher flexibility of attribute-based approaches comes along with higher complexity in the specification and maintenance of the policies. A policy administrator needs to be aware of the attribute scheme used by the issuer of a specific attribute, who in many cases may reside in a different organization.

With the background of a more and more developing Semantic Web, we propose to put user, resource, and environment attributes into a semantic context, simplifying the specification and maintenance of policies. As the Internet is the main application area of our approach, we build upon standards like XACML [6], RDF [12], and OWL [14].

The remainder of this paper is organized as follows: Sections 2 and 3 introduce the basics regarding attribute-based access control and the Semantic Web respectively. In section 4 we present an extension to the XACML architecture that allows easier policy specification by introducing the use of ontologies. A prototype implementation is covered in section 5. Section 6 describes related approaches found in the literature and compares them to ours. Section 7 concludes the paper and discusses possible future work.

2. Attribute-based Authorization and Access Control

The basic idea of attribute-based access control (ABAC) [7, 8] is not to define permissions directly between subjects and objects, but instead to use their attributes as the basis for authorizations. For subjects, attributes can be static ones like a subject's position or role in a company. However, dynamic attributes like age, current location or an acquired subscription for a digital library can be used as well. For objects, metadata properties, e.g., the subject of a document, can be used. This concept is illustrated in figure 1.

Subjects and objects are both represented by a set of attributes and related attribute values. Permissions consist of the combination of a so-called object descriptor, which consists of a set of attributes and conditions like "age > 18" or

Proceedings of the First International Conference on Availability, Reliability and Security (ARES'06)
0-7695-2567-9/06 $20.00 © 2006 IEEE

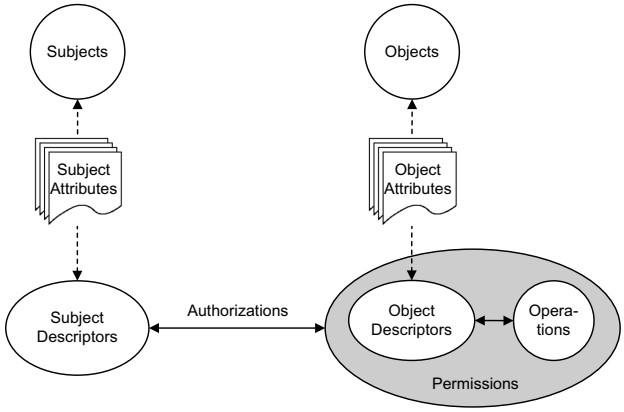

Figure 1. Overview of the ABAC model

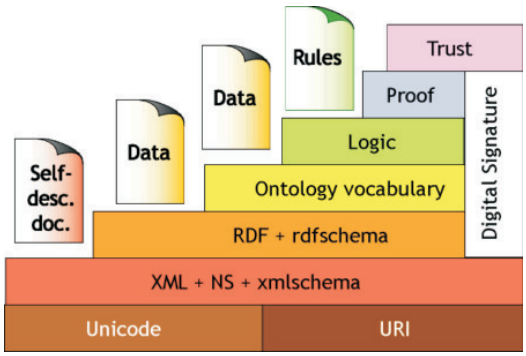

Figure 2. Layers of the Semantic Web

"subscribed = TRUE", and an operation that is to be executed on the objects denoted by the descriptor. Authorizations are defined between a subject descriptor and a permission. They can be extended with additional conditions, e.g. for comparing subject attributes with object attributes (without conditions, attributes can only be compared with constant values). Besides, environment attributes like time of day can be considered for the access control decision [8].

Like mentioned before, attribute-based approaches are particularly suitable for authorization and access control in open and distributed systems. These systems are however highly heterogeneous and so are the attribute schemes that are potentially used. The attributes a user possesses do not necessarily match those used by the developers of a Web-based information system or service. For instance, an e-commerce platform could represent adult customers by an attribute "fullAge", whereas customers try to prove this with attributes like "age" or by providing a driver's license ("hasDriversLicense"). In classic attribute-based approaches the policy administrator has to consider this already in advance, which significantly complicates the management of ABAC policies.

Thus we argue that policy specification can be simplified by introducing semantic inference at the point of the access control decision. For this purpose we extend the open and widely accepted XACML standard [6]. XACML is an XML dialect for defining attribute-based access control policies. It is well-suited for our purposes because all relevant aspects of the introduced ABAC model can be represented.

3. Ontologies and Semantic Web Technologies

In 1998, Tim Berners-Lee presented his roadmap towards the Semantic Web [1]. His main idea was to enrich the human-readable information on the Web with metadata with precise semantic. A standardized metadata model, a standardized syntax, and the possibility to define a standardized vocabulary (a so-called ontology) was needed. Figure 2 shows an often used diagram which presents the technologies developed within the Semantic Web initiative of the World Wide Web Consortium (W3C)[1] as a layer model.

Resources on the Semantic Web contain metadata. Using an XML syntax, this metadata is encoded using the Resource Description Framework (RDF) [12] as the language for the description of resources and RDF Schema (RDFS) [13] for the definition of the metadata schema. RDFS is based on a class concept and inheritance and and already features simple constructs for describing ontologies. An ontology is capable of describing concepts, e.g., persons, that exist in a certain domain and relationships among them. These concepts can then also be used as a vocabulary for the metadata of resources. The standard for describing ontologies on the Semantic Web is the Web Ontology Language (OWL) [14], supporting richer semantics on top of RDFS. The processing and analysis of ontologies, i.e. drawing conclusions and gaining new information through combination, takes place in the logical layer. Implicit information in the data can be made explicit by using so-called reasoners or inference engines. Simple inferences are already possible with RDFS and OWL, for instance through inheritance. More complex custom inference rules require the usage of a special rule language. A promising approach based on the Rule Markup Language (RuleML) is the Semantic Web Rule Language (SWRL) [2].

Meanwhile integrated tools for managing RDF(S) and OWL data are available (e.g., Jena[2] and Sesame[3]). Beside persistent storage, they also provide simple inference capabilities as well as query languages like SPARQL [15]. The remaining layers of the Semantic Web, particularly proof and trust, are so far still much of a field of research. The ap-

1 http://www.w3.org/2001/sw/
2 http://jena.sourceforge.net
3 http://www.openrdf.org

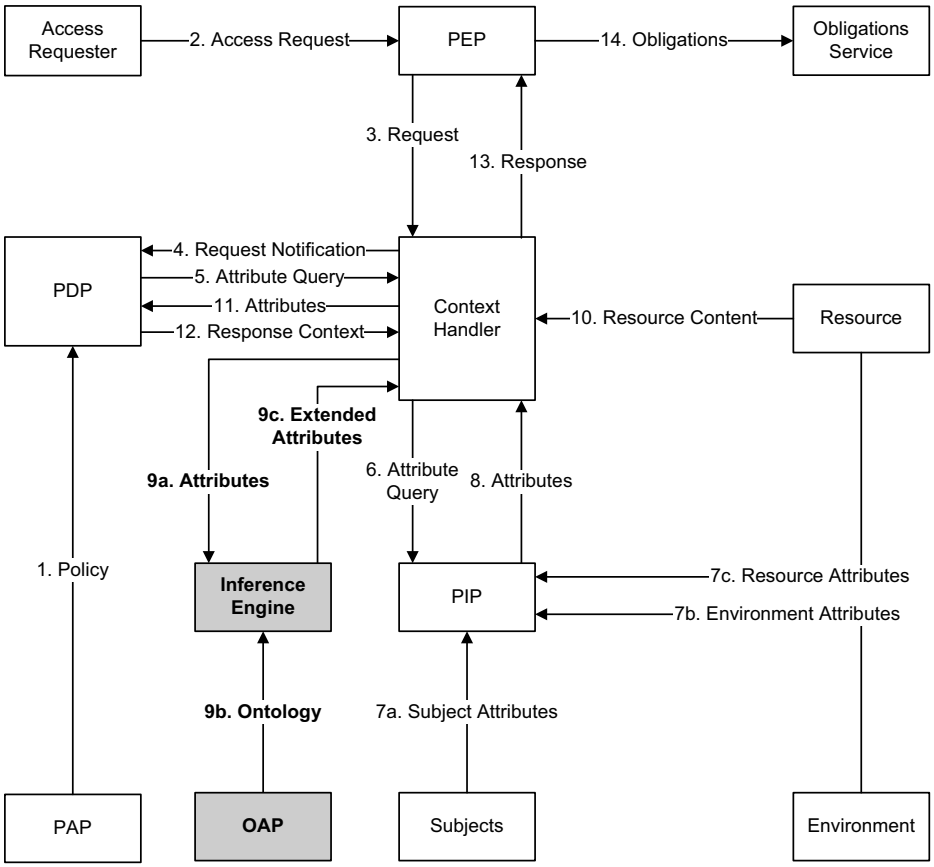

Figure 3. Extended XACML architecture

proach for ontology-based authorization and access control presented in the following section is based on the standards OWL, SWRL and SPARQL.

4. Proposed Architecture

For the attribute-based access control techniques at hand, e.g., XACML, all user attributes and conditions we want to check (e.g., "age > 18") must be encoded statically in the policy. Thus, complexity rises enormously and maintenance and management gets more error-prone. Actually, only the conditions necessary from the system designer's point of view (e.g., "fullAge = true") should be relevant when specifying the policies. It is rather a question of attribute management how a user can prove this property.

Our approach achieves this by extending the architecture defined in the XACML specification [6] with Semantic Web techniques. Mappings between different attributes and attribute conditions are performed by an ontology-based inference engine. The extended XACML architecture is depicted in figure 3. Our extensions are emphasized using grey shading and bold labels.

An access control decision and enforcement is now performed according to the following sequence:

1. A so-called policy administration Point (PAP) creates a XACML policy and provides it to the policy decision point (PDP). Figure 7 at the end of this paper shows an example for such a policy. In our example, we assume users have to be adult when requesting read access to resources below `http://www.example.org/restricted/`.
2. The user (access requester) sends a resource request to the policy enforcement point (PEP), e.g., for `http://www.example.org/restricted/index.html`.
3. The PEP forwards this request (which may already contain user, resource, and environment attributes) to the context handler. It is assumed that the user is identified by his email address `user@example.org` and that he has included an "age" attribute.
4. The context handler creates a XACML request and sends it to the PDP.

5. In case the PDP needs additional subject, resource, and environment attributes, they are requested from the context handler. In this example, the PDP is in need of the attribute "fullAge".
6. The context handler requests those attributes from a policy information point (PIP).
7. The PIP collects the requested attributes, if possible, from the subject, resource, and environment. In this case, there is no possibility to acquire the attribute "fullAge" from the user.
8. The PIP delivers the attributes back to the context handler.
9. So far, the procedure was exactly as specified in the XACML standard. If some attributes requested by the PDP still are missing, we propose to try to deduce them from the ones included in the request and supplied by the PIP, utilizing Semantic Web technologies. The attributes are sent to an inference engine (step 9a). The inference engine combines the attributes with an ontology delivered by an ontology administration point (OAP). Figure 8 at the end of this paper shows a sample ontology in OWL/XML syntax. The example includes a declaration that the attribute "fullAge" is derived from "hasDriversLicense" (i.e. everyone who has a driver's license is of full age) and an SWRL rule that defines full age depending on the "age" attribute:

   ```
   Subject(?x) ∧ age(?x, ?a) ∧
   greaterThanOrEqual(?a, 18)
   ⇒ fullAge(?x, true)
   ```

   The derived attributes can then be queried by the context handler with a SPARQL request using the DESCRIBE command (step 9b), for instance:

   ```
   DESCRIBE <mailto:user@example.org>
   ```

   The inference engine delivers the complete set of attributes as shown in figure 4 (step 9c); the derived attribute "fullAge" is shown boldly.
10. The further processing again corresponds to the XACML specification . Optionally, the context handler attaches the resource itself to the request.
11. The extended request is sent to the PDP. Figure 5 shows the complete sample request (with derived attributes) in XACML syntax. The derived attribute "fullAge" is again shown in bold.
12. The PDP evaluates the policy and sends the response context (including the access control decision) back to the context handler.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf=
  "http://www.w3.org/1999/02/22-rdf-syntax-ns#">

<Subject
  xmlns="urn:oasis:names:tc:xacml:1.0:context:"
  rdf:about="mailto:anonymous@anonymous.org">
  <age xmlns="urn:example:" rdf:datatype=
    "http://www.w3.org/2001/XMLSchema#integer"
    >30</age>
  <fullAge xmlns="urn:example:" rdf:datatype=
    "http://www.w3.org/2001/XMLSchema#boolean"
    >true</age>
</Subject>

...

</rdf:RDF>
```

Figure 4. Extended subject attribute set in RDF/XML

13. The context handler translates the response context back to the native format of the PEP and forwards it.
14. The PEP satisfies possible obligations.
15. (Not shown) If access is granted, the PEP allows access to the resource. Otherwise, access is refused.

5. Prototype Implementation

In order to evaluate our proposal, we have implemented a prototype that follows the architecture presented in the previous section with minor simplifications. As the source of the attributes is irrelevant for their further processing, we did not use a policy information point (PIP). Instead we assume that all available attributes are already included in the request. Also, the actual resource and a policy enforcement point (PEP) are omitted in our test scenario. Hence, the original request is already in XACML format, rather than having a native PEP request. Furthermore, we do not consider obligations in our prototype because they are not associated with the attribute management, either.

The architecture of our prototype implementation is depicted in figure 6. The access control decision procedure is similar to section 4. After having received a XACML request, the context handler extracts the attributes from the request, reproduces them in RDF and hands them over to the inference engine. For this purpose, we use Jena by Hewlett Packard[4] (version 2.2) which combines the attributes with an OWL ontology. Unfortunately, in its current version Jena does not support custom rules using SWRL. However, deduction is possible via inheritance or OWL constructs like "owl:equivalentProperty" and "owl:sameAs". The complete

4 http://jena.sourceforge.net

```
<?xml version="1.0"?>
<Request xmlns="urn:oasis:names:tc:xacml:1.0:context">

<Subject>
  <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
    DataType="urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name">
    <AttributeValue>user@example.org</AttributeValue>
  </Attribute>
  <Attribute AttributeId="urn:example:age" DataType="http://www.w3.org/2001/XMLSchema#integer">
    <AttributeValue>30</AttributeValue>
  </Attribute>
  <Attribute AttributeId="urn:example:fullAge" DataType="http://www.w3.org/2001/XMLSchema#boolean">
    <AttributeValue>true</AttributeValue>
  </Attribute>
</Subject>

<Resource>
  <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
    DataType="http://www.w3.org/2001/XMLSchema#string">
    <AttributeValue>http://www.example.org/restricted/index.html</AttributeValue>
  </Attribute>
</Resource>

<Action>
  <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
    DataType="http://www.w3.org/2001/XMLSchema#string">
    <AttributeValue>read</AttributeValue>
  </Attribute>
</Action>

</Request>
```

Figure 5. Extended request context in XACML

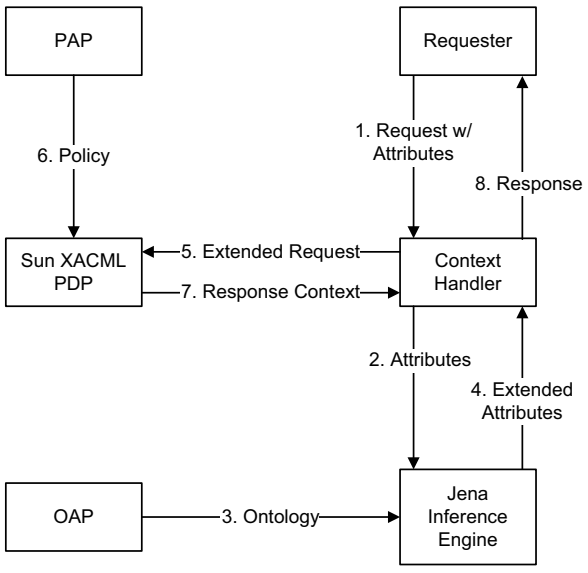

Figure 6. Architecture of the prototype implementation

(extended) attribute set is requested by the context handler from Jena using SPARQL and incorporated into the XACML request. We use the XACML reference implementation by Sun Microsystems[5] (version 1.2) as the policy decision point (PDP).

6. Related Work

A first approach that employs Semantic Web technologies in the access control decision and enforcement process is by Yagüe et al. [11]. Their proposal defines an own policy language on the basis of XML. It also allows for dynamic instantiation, i.e. querying external XML and RDF data sources for attribute values. However, a more powerful metadata representation like OWL and reasoning capabilites are not provided.

The KAoS framework [10] is a collection of services for distributed policy management and enforcement. It uses OWL ontologies to specify also the policies themselves. This can pose difficulties because the gap between specification and actual implementation of such policies cannot be coped with automatically [9]. Likewise, the policy engine in Rei [5] can handle policies specified in RDFS or a special logic programming language. As our aim is to simplify attribute and policy management we stick with the es-

5 http://sunxacml.sourceforge.net

tablished policy language XACML and extend it by the use of OWL for attribute management.

Damiani et al. [3] use XACML as their policy language. They extend it with an operator to trigger requests for object metadata from a semantic environment. Subject metadata is used for the access control decision as delivered by the requester. Our approach is different in two ways. Firstly, the policy administrator does not have to explicitly include a semantic request in each policy. Secondly, it is possible to reason also about subject metadata in our proposal.

7. Conclusions

We have presented an approach for simplifying the specification and maintenance of attribute-based access control policies by extending the attribute management with an ontology-based inference facility. This enables policy administrators to concentrate on the properties they deem necessary from their point of view; they do not need to determine in advance which attributes a subject may use to prove these properties. A semantic mapping between different attributes can be performed in an ontology. Our proposal is based on the established XACML standard and features thorough use of open standards like RDF and OWL in the semantic extension of the architecture. The widespread use of XACML will also ensures the availability of powerful and easy-to-use policy editors in the near future.

On first sight it might seem that the simplified policy management has been traded for a rather complicated ontology/attribute management. However, with the further development of the Semantic Web in various areas, it can be expected that more and more pre-built ontologies will become available, e.g., for e-government. These can be provided by corporate, national, or even international organizations, but also by software providers that employ attribute-based access control technology. Policy administrators will be able to reuse such ontologies for different authorization scenarios. Hence, the ontology has to be built only once.

We plan to evaluate the benefits of the presented approach in future projects. One step is the integration of the presented implementation into a knowledge portal prototype with a content management facility. Furthermore, we will explore the applicability of the approach in the area of (semantic) web services in the field of e-government.

References

[1] Berners-Lee, T.: A Roadmap to the Semantic Web. World Wide Web Consortium, September 1998. http://www.w3.org/DesignIssues/Semantic.html

[2] SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Joint US/EU ad hoc Agent Markup Language Committee, November 2003. http://www.daml.org/2003/11/swrl/

[3] Damiani, E., De Capitani di Vimercati, S., Fugazza, C., Samarati, P.: Extending Policy Languages to the Semantic Web. *Proc. Web Engineering - 4th International Conference (ICWE 2004)*, Munich, Germany, July 2004.

[4] Ferraiolo, D.F., Sandhu, R., Gavrila, S., Kuhn, D., and Chandramouli, R.: Proposed NIST Standard for Role-based Access Control. *ACM Transactions on Information and Systems Security, Volume 4, Number 3*, August 2001.

[5] Kagal, L., Finin, T., Joshi, A.: A Policy Based Approach to Security for the Semantic Web. *Proc. 2nd International Semantic Web Conference (ISWC 2003)*, Sanibel Island, FL, October 2003.

[6] OASIS eXtensible Access Control Markup Language Technical Committee: eXtensible Access Control Markup Language (XACML). http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml

[7] Priebe, T., Fernandez, E.B., Mehlau, J.I., Pernul, G.: A Pattern System for Access Control. *Proc. 18th Annual IFIP WG 11.3 Working Conference on Data and Application Security*, Sitges, Spain, July 2004.

[8] Priebe, T., Dobmeier, W., Muschall, B., Pernul, G.: ABAC – Ein Referenzmodell für attributbasierte Zugriffskontrolle. *Proc. 2. Jahrestagung Fachbereich Sicherheit der Gesellschaft für Informatik (Sicherheit 2005)*, Regensburg, Germany, April 2005.

[9] 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. *Proc. 2nd International Semantic Web Conference (ISWC 2003)*, Sanibel Island, FL, October 2003.

[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. *Proc. 4th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2003)*, Comersee, Italy, June 2003.

[11] Yagüe, M., Mana, A., Lopez, L., Troya, J.M.: Applying the Semantic Web Layers to Access Control. *Proc. of the DEXA 2003 Workshop onWeb Semantics (WebS 2003)*, Prague, Czech Republic, September 2003.

[12] Resource Description Framework (RDF): Concepts and Abstract Syntax. World Wide Web Consortium, February 2004. http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/

[13] RDF Vocabulary Description Language 1.0: RDF Schema. World Wide Web Consortium, February 2004. http://www.w3.org/TR/2004/REC-rdf-schema-20040210/

[14] OWL Web Ontology Language Overview. World Wide Web Consortium, February 2004. http://www.w3.org/TR/2004/REC-owl-features-20040210/

[15] SPARQL Query Language for RDF. World Wide Web Consortium, February 2005. http://www.w3.org/TR/2005/WD-rdf-sparql-query-20050217/

```
<?xml version="1.0"?>
<Policy xmlns="urn:oasis:names:tc:xacml:1.0:policy"
  PolicyId="SamplePolicy" RuleCombiningAlgId=
  "urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:permit-overrides">

<Target>
  <Subjects>
    <AnySubject/>
  </Subjects>
  <Resources>
    <Resource>
      <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:regexp-string-match">
        <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string"
          >http://www.example.org/restricted/.*</AttributeValue>
        <ResourceAttributeDesignator DataType="http://www.w3.org/2001/XMLSchema#string"
          AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"/>
      </ResourceMatch>
    </Resource>
  </Resources>
  <Actions>
    <AnyAction/>
  </Actions>
</Target>

<Rule RuleId="SampleRule" Effect="Permit">
  <Target>
    <Subjects>
      <Subject>
        <SubjectMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
          <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#boolean"
            >true</AttributeValue>
          <SubjectAttributeDesignator AttributeId="urn:example:fullAge"
            DataType="http://www.w3.org/2001/XMLSchema#boolean"/>
        </SubjectMatch>
      </Subject>
    </Subjects>
    <Resources>
      <AnyResource/>
    </Resources>
    <Actions>
      <Action>
        <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
          <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string"
            >read</AttributeValue>
          <ActionAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
            DataType="http://www.w3.org/2001/XMLSchema#string"/>
        </ActionMatch>
      </Action>
    </Actions>
</Rule>

...

</Policy>
```

Figure 7. Sample policy in XACML

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:swrl="http://www.w3.org/2003/11/swrl#">

<owl:Class rdf:about="urn:oasis:names:tc:xacml:1.0:policy:Subject"/>

<owl:DatatypeProperty rdf:ID="urn:example:age">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"/>
  <rdfs:domain rdf:resource="urn:oasis:names:tc:xacml:1.0:policy:Subject"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="urn:example:hasDriverLicense">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#boolean"/>
  <rdfs:domain rdf:resource="urn:oasis:names:tc:xacml:1.0:policy:Subject"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:about="urn:example:fullAge">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#boolean"/>
  <rdfs:domain rdf:resource="urn:oasis:names:tc:xacml:1.0:policy:Subject"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
  <rdfs:subClassOf rdf:resource="urn:example:hasDriverLicense"/>
</owl:DatatypeProperty>

<swrl:Variable rdf:ID="x"/>
<swrl:Variable rdf:ID="a"/>
<swrl:Imp>
  <swrl:body rdf:parseType="Collection">
    <swrl:ClassAtom>
      <swrl:classPredicate rdf:resource="urn:oasis:names:tc:xacml:1.0:policy:Subject"/>
      <swrl:argument1 rdf:resource="#x"/>
    </swrl:ClassAtom>
    <swrl:DatavaluedPropertyAtom>
      <swrl:propertyPredicate rdf:resource="urn:example:age"/>
      <swrl:argument1 rdf:resource="#x"/>
      <swrl:argument2 rdf:resource="#a"/>
    </swrl:DatavaluedPropertyAtom>
    <swrl:BuiltinAtom>
      <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#greaterThanOrEqual"/>
      <swrl:argument1 rdf:resource="#a"/>
      <swrl:argument2 rdf:datatype="http://www.w3.org/2001/XMLSchema#integer"
        >18</rdf:argument2>
    </swrl:BuiltinAtom>
  </swrl:body>
  <swrl:head rdf:parseType="Collection">
    <swrl:DatavaluedPropertyAtom>
      <swrl:propertyPredicate rdf:resource="urn:example:fullAge"/>
      <swrl:argument1 rdf:resource="#x"/>
      <swrl:argument2 rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
        >true</swrl:argument2>
    </swrl:DatavaluedPropertyAtom>
  </swrl:head>
</swrl:Imp>

...

</rdf:RDF>
```

Figure 8. Sample ontology in OWL/XML