Supporting Attribute-based Access Control in Authorization and Authentication Infrastructures with Ontologies

Torsten Priebe
Capgemini Consulting Österreich AG, Vienna, Austria
Email: torsten.priebe@capgemini.com

Wolfgang Dobmeier, Christian Schläger
Department of Information Systems, University of Regensburg, Germany
Email: {wolfgang.dobmeier, christian.schlaeger}@wiwi.uni-regensburg.de

Nora Kamprath
KPMG Deutsche Treuhand-Gesellschaft Aktiengesellschaft, Frankfurt/Main, Germany
Email: nkamprath@kpmg.com

***Abstract*— In highly open systems like the Internet, attribute-based access control (ABAC) has proven its appropriateness. This is reflected in the utilization of ABAC in authentication and authorization infrastructures (AAIs). However, specification and maintenance of ABAC policies has turned out to be complex and error-prone even in federations of limited size, especially if heterogeneous attribute schemes are involved. Here, the arising Semantic Web can contribute to a solution. This paper describes an architecture for embedding the access control process into a semantic context employing external knowledge in form of ontologies. We base our proposal on extensions of established open standards. Using the approach presented, policy management at the different sites of a federation is simplified by a semantic attribute management facility.**

***Index Terms*—Security, attribute-based access control, authorization and authentication infrastructures, attribute management, semantic web, ontologies.**

I. 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) [1] were used in such environments. 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) [2], [3], [4] has been introduced, e.g., within the eXtensible Access Control Markup Language (XACML) standard [5]. However, the higher flexibility of attribute-based approaches comes along with higher complexity in the specification and maintenance of the policies. Policy administrators need 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. The attributes a user possesses do not necessarily match those used by the developers of a web-based information system or service.

Besides, infrastructures have been developed supporting the exchange of security information and user attributes as well as performing authorization and authentication tasks in federations; we call these infrastructures authentication and authorization infrastructures (AAIs). Among the solutions in use, the best known ones are Liberty's ID-FF, Internet2's Shibboleth, and Microsoft's .NET Passport. Parties like enterprises and e-commerce vendors can use AAIs to form federations and share user and security information for authentication and authorization purposes like it has been shown in [14]. Being part of such a federation in connection makes policy management even more difficult.

We use an example from e-commerce to illustrate this. A video-on-demand store could represent adult customers by an attribute "fullAge" in his access control policy. Furthermore, a book vendor who is part of the same federation could describe adult customers using the attribute "age". On the other hand, a potential customer might try to prove these properties by

Based on "Supporting Attribute-based Access Control with Ontologies", by Torsten Priebe, Wolfgang Dobmeier, and Nora Kamprath which appeared in the Proceedings of the First International Conference on Availability, Reliability and Security (ARES 2006), Vienna, Austria, April 2006.

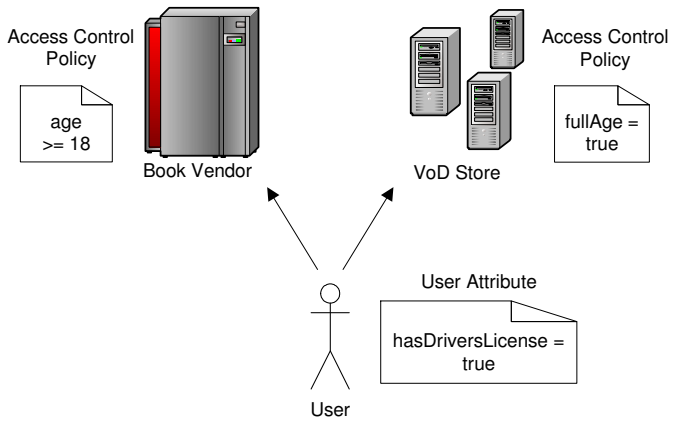

Figure 1. Sample e-commerce scenario

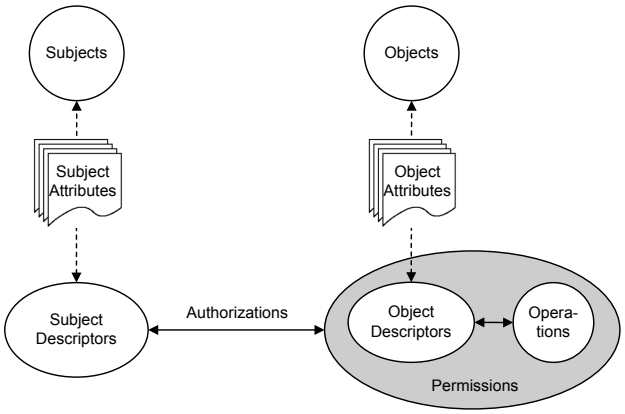

Figure 2. Overview of the ABAC model

providing a driver's license ("hasDriversLicense") which can be valid for many countries. This scenario is depicted in Fig. 1.

In classic attribute-based approaches the policy administrators at the different sites have to consider this situation already in advance, which significantly complicates the management of ABAC policies. On the other hand, if the different organizations restrict themselves to a common set of standardized attributes, this would happen at the price of a low expressiveness for the representation of subjects and objects, thus losing some of the advantages of the flexible and dynamic ABAC functionality.

In this paper, we propose a solution to these issues. With the background of a more and more developing Semantic Web, we propose to put user, resource, and environment attributes into a semantic context as it has been initially described in [6]; this will simplify the specification and maintenance of policies. We then transfer this solution into an AAI scenario, showing how our proposal can be incorporated into current AAI protocols. As the Internet is the main application area of our approach, we build upon standards like XACML [5], RDF [7], and OWL [8].

The paper is organized as follows: In the next section we introduce the concepts of attribute-based access control and XACML in more detail. Section III describes authentication and authorization infrastructures, while section IV presents basics of the Semantic Web. In section V we propose an extension to the XACML architecture that allows easier policy specification by introducing the use of ontologies and then apply it to a generic AAI scenario. A prototype implementation is covered in section VI. An in-depth discussion of our approach is given in section VII. In section VIII, a description of related approaches found in the literature follows. Section IX concludes the paper and discusses possible future work.

II. Attribute-based Authorization and Access Control

The basic idea of attribute-based access control (ABAC) [2], [3] 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 name, or 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 Fig. 2.

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 "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. Using descriptors it is possible to dynamically assign permissions to subjects and objects, thus making a manual assignment superfluous.

In addition, authorizations 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). Environment attributes like time of day can also be considered for the access control decision. Furthermore, it has been shown that traditional access control models like discretionary, mandatory, and role-based access control can be mapped to ABAC concepts. [3]

As described in the introduction, we argue that policy specification for ABAC-like authorization models is complex and error-prone. It 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 [5]. XACML defines a generic authorization architecture as well as an XML dialect for specifying attribute-based access control policies. It is well-suited for our purposes because all relevant aspects of the introduced ABAC model can be represented.

A XACML authorization architecture consists of several logical components. First of all, a reference monitor concept is used to intercept access requests. This component is called a Policy Enforcement Point (PEP). A PEP transmits access requests to a Policy Decision Point (PDP) for retrieval and evaluation of applicable policies. Policies are specified and stored in Policy Administration Points (PAPs). In case a PDP needs attributes of subjects, objects, or the environment that are missing in the original request, Policy Information Points

(PIPs) deliver the data needed for evaluation. As in general not all of these components will use the same message format for communication, a Context Handler is employed as a mediator. After evaluation, the result is sent back to the requester and is enforced at the PEP. One or more obligations can be executed then, e.g. creating a log entry or sending an email.

As mentioned, the XACML specification furthermore provides a policy language and a request/response language. The request/response language defines the declaration of access control requests for a specific object and responses to these requests. The policy language is based on three main elements: PolicySet, Policy, and Rule. A PolicySet can contain a set of single policies or another PolicySet. Policies consist of single Rules which have a Condition, an Effect, and a Target. Conditions can be used beyond the Target to further specify the applicability of a Rule using predicates, while Effects denote the result of a Rule, e.g. permit or deny. To find the relevant policy for an access control request, every PolicySet, Policy, and rule has a Target, which is evaluated at the time of access request. A Target consists of a specification of sets of subjects, objects, operations, and environments using their respective attributes which can be evaluated with match functions. When the relevant Policies and Rules are found, they are evaluated independently of each other; contradicting evaluation results can be resolved using rule- as well as policy-combining algorithms.

III. Authentication and Authorization Infrastructures

Service providers on the Internet are familiar with infrastructures providing basic security services. Authentication and authorization infrastructures (AAIs) have started with basic single sign-on functionality and are nowadays also able to manage the authorization process and access control decisions. Two main architectures can be found: central ones like Microsoft's .NET Passport solution[1] or federated ones like Liberty's Identity Federation Framework (Liberty ID-FF)[2] or Internet2's Shibboleth[3].

Traditionally, AAIs consist of one or more identity provider (IdP), authenticating users, numerous service providers (SP), offering their services backed by the AAI, and potentially additional AAI providers, contributing services like storage, certificate management, auditing, and likewise. Centrally organized AAIs usually rely on a third party provider that offers his security service to SPs (MS Passport approach). In a federation, SPs are joining together, trying to exploit synergies. Each federation member offers services to the others (Liberty ID-FF approach).

AAIs help sharing security information about subjects and objects with other service providers or central services. Such information could be for example an assertion about the user's correct and traceable identification. Additionally, trusted sources can provide profile information like user attributes. Using these attributes an access control model like ABAC (e.g., using XACML) can decide on the user's rights. Fig. 3 shows the process of granting access to resources with the help of user and resource attributes. Attributes can contain identity and profile information. An AAI takes over various elements of the shown security services. E.g., if the infrastructure federates the user authentication the result is a single sign-on (SSO) for the user.

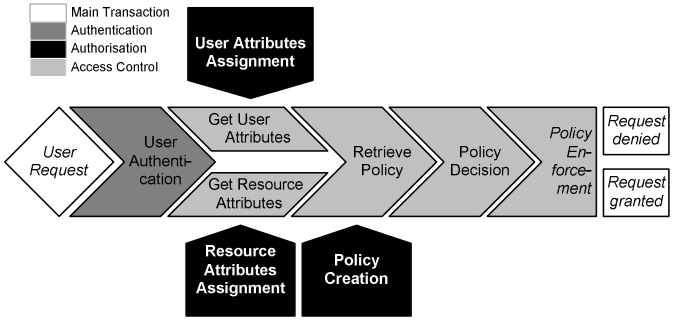

Figure 3. Security services including an attribute infrastructure

A special benefit of AAIs lies in the accumulated user data over a federation: user profiles, buying patterns, and earned privileges. Identities could be transferred from one service provider to another making it possible to always use up-to-date address data or prove a good reputation acquired at one federation member. The exchange of attributes can be done via the Security Assertion Markup Language (SAML). As an open standard it is predetermined to form the underlying communication technology for every open AAI approach. Its potential to sign, encrypt, and communicate any kind of attributes builds also the basis for XACML decisions. SAML tokens can be transferred via multiple means; HTTP Post-requests and Web Services are the prevailing forms.

All AAI solutions analyzed have SSO functionality in common. However, neither of them is able to provide attribute-based access control. At most, attribute exchange is supported. The implementation of SAML within Liberty's ID-FF and Shibboleth is exemplary for an open AAI.

Including ABAC functionality and an attribute infrastructure into AAIs normally leads to the restriction or centralization of attributes. The heterogeneity of user information at each federation member can only be solved by a centrally maintained database containing all attributes and XACML policies or via a limited, common set of attributes. The centralized approach has been discussed in [9].

In a generic AAI with centralized attribute management the communication steps are as shown in Fig. 4. In step 1 the user requests the resource he desires from any service provider (SP). As the SP has delegated the security services to the AAI the user is referred to a central AAI server requesting an access control decision for the specific resource. First, the AAI authenticates the user appropriately (step 3). Once the user is identified the AAI collects his attributes from the central database and uses the access policy to decide on the request (steps 4-6). The policy decision is referred to the SP which enforces the decision. After the transaction the SP can update the customer's attributes if necessary. Please note that the user

[1] http://www.passport.net
[2] http://www.projectliberty.org
[3] http://shibboleth.internet2.edu

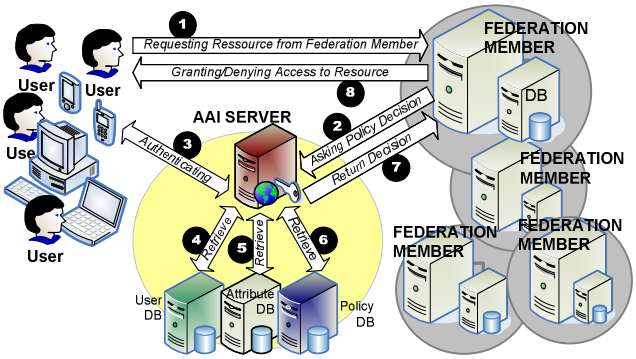

Figure 4. ABAC enabled AAI with centralized attribute management

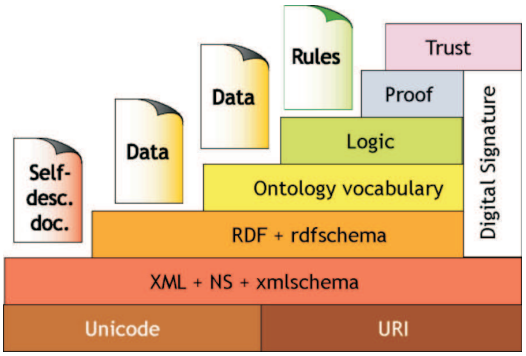

Figure 5. Layers of the Semantic Web

data is not stored at the SP but at a central AAI facility. Furthermore, all federation members need to use the same attribute format and information.

IV. Ontologies and Semantic Web Technologies

In 1998, Tim Berners-Lee presented his roadmap towards the Semantic Web [10]. His main idea was to enrich the human-readable information on the Web with metadata with precise semantics. A standardized metadata model, a standardized syntax, and the possibility to define a standardized vocabulary (a so-called ontology) was needed. Fig. 5 shows an often used diagram which presents the technologies developed within the Semantic Web initiative of the World Wide Web Consortium (W3C)[4] 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) [7] as the language for the description of resources and RDF Schema (RDFS) [11] for the definition of the metadata schema. RDFS is based on a class concept and inheritance 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) [8], 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) [12].

Meanwhile integrated tools for managing RDF(S) and OWL data are available (e.g., Jena[5] and Sesame[6]). Besides persistent storage, they also provide simple inference capabilities as well as query languages like SPARQL [13]. The remaining layers of the Semantic Web, particularly proof and trust, are so far still much of a field of research. The approach for ontology-based authorization and access control presented in the following section is based on the standards OWL, SWRL and SPARQL.

V. Proposed Architecture

In this section we introduce our proposal for putting the access control process into a semantic context by utilization of knowledge encoded in an ontology. The description is split into two parts. First, we present the core of our approach which extends the process of gathering user attributes in XACML by attributes which are inferred employing Semantic Web technologies. Second, we embed this extension into a generic AAI architecture; the goal is to facilitate policy specification in a defined federation of sites by deploying our attribute management facility.

A. Basic Approach

For the attribute-based access control techniques at hand, e.g., XACML, all user attributes and conditions to be checked (e.g., “age > 18”) must be encoded statically in the policy. 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 [5] 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 Fig. 6. Our extensions are emphasized using gray shading and bold labels.

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

1) The PAP provides a XACML policy, which has been created beforehand by a security administrator, to the PDP. Fig. 12 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, e.g., for

[4] `http://www.w3.org/2001/sw/`

[5] `http://jena.sourceforge.net`

[6] `http://www.openrdf.org`

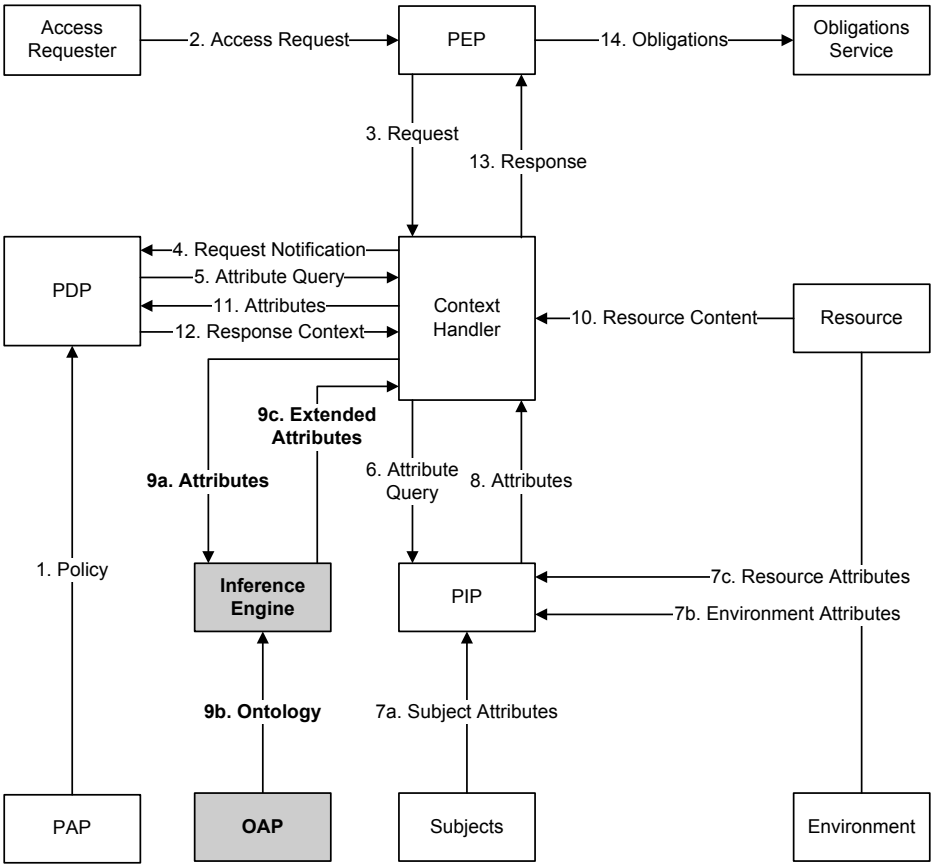

Figure 6. Extended XACML architecture

`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.
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). Fig. 13 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 Fig. 7 (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. Fig. 8 shows the complete sample request (with derived attributes) in

```
<?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 8. Extended request context in XACML

```
<?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 7. Extended subject attribute set in RDF/XML

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.
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.

B. Embedding the Extension Into a Generic AAI Architecture

Including our proposal into the presented AAI leads to an extension of the introduced service chain as shown in Fig. 9. After retrieving attributes and policy the proposed inference

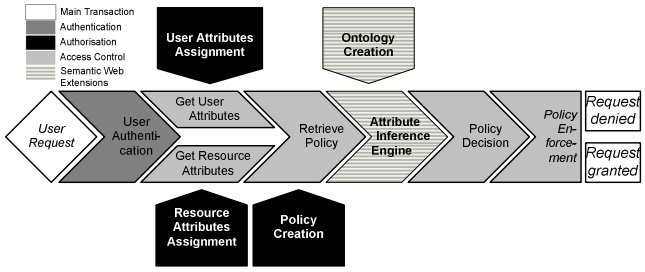

Figure 9. Security services enhanced with attribute inference engine

engine needs to be contacted. The engine is able to deduce new attributes or map attributes from different providers. As already explained, user attributes now can be maintained, stored, and formatted to every federation member’s liking. This enables a flexible and dynamic attribute exchange in an open federation. With the semantic mapping of user information the need for a centralised managed repository evaporates.

In [14] we have shown a generic architecture for a distributed AAI making use of distributed identity providers (IdP) as used in Shibboleth or Liberty and one or more PDPs responsible for collecting attributes and computing a policy decision. So far the potential attributes to use needed to be specified centrally for each federation leading to the already given drawbacks.

Using the proposed XACML enhancements these limitations can be avoided. The result of a distributed AAI with included ontology administration point (OAP) and inference mechanism is shown in Fig. 10.

We made use of the main elements of Liberty ID-FF 1.1, namely distributed identity and service providers, and used

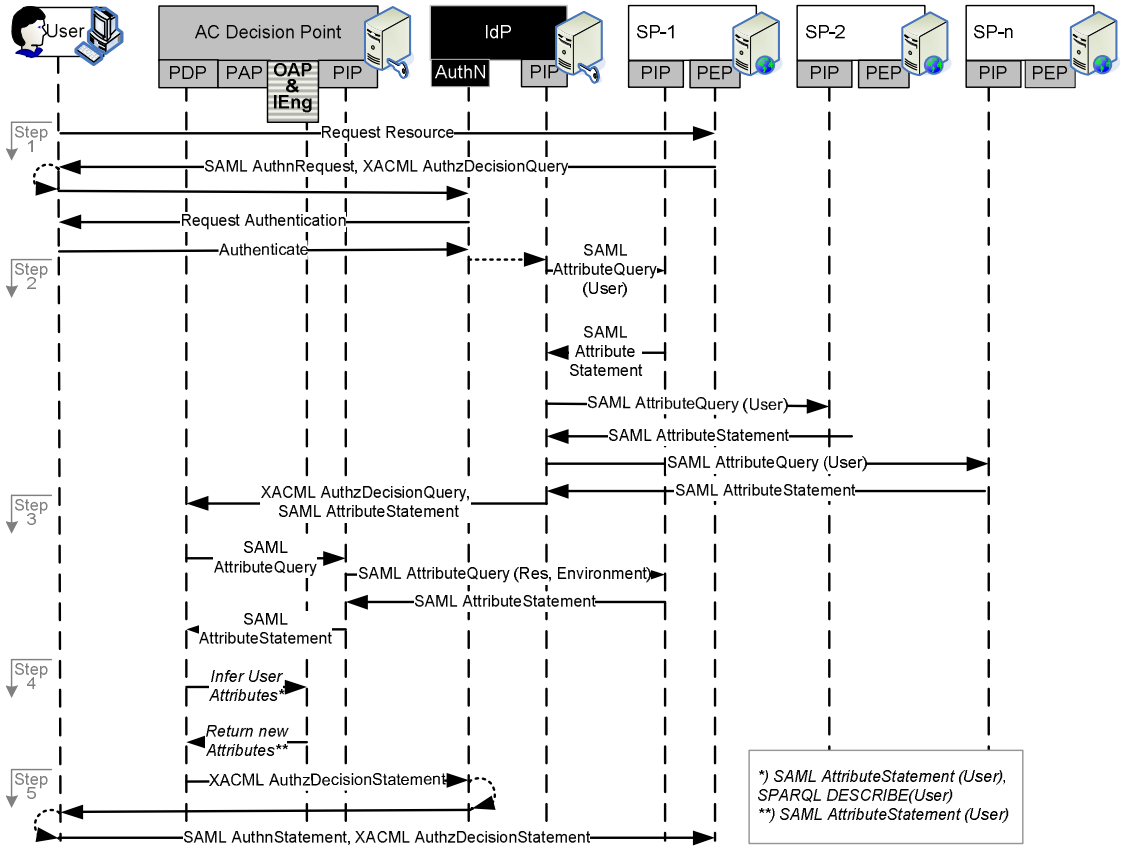

Figure 10. ABAC-enabled AAI with inferred attributes

SAML 1.0. The first step of the access control decision, the authentication, is handled as defined by the Liberty ID-FF protocol. In addition to these parties we introduce the XACML elements policy information point (PIP), policy administration point (PAP), policy decision point (PDP), policy enforcement point (PEP), the described ontology administration point (OAP), and inference engine (IEng). As the XACML standard is very imprecise about the PIP and the communication between other XACML elements and the PIP we decided on using the PIP concept mainly as an interface to relevant databases and a transformation tool of information and attributes into SAML. The originally proposed context handler has been substituted by this functionality. The sequence diagram in Fig. 10 is based on the the UML 2.0 sequence diagram notation using SAML and XACML nomenclature.

Access control decision and enforcement is now performed according to the following steps:

1) When the user tries to access a resource he is referred to his IdP. The IdP is derived either from a cookie stored in the user's browser or the user chooses from a list. This is the standard Liberty ID-FF SSO procedure. The SP (SP-1 in Fig. 4) sends the IdP a SAML authentication request. Additionally, he sends a XACML authorization decision request.
2) After the user's authentication, the IdP's PIP component collects the available attributes. He asks every federation member with a SAML request if attributes are available. The service providers' PIPs provide the IdP with SAML AttributeStatements.
3) After the IdP's PIP has collected all attributes, an access control decision request is sent to the PDP with an identifier of the requested resource and all user attributes. The PDP's PIP will collect resource and environment attributes.
4) In accordance with step 9 explained in section V-A the PDP will try to deduce attributes from the information at hand. The user attributes are sent to the inference engine together with a SPARQL DESCRIBE request for further information. Using the given ontology from the OAP, the inference engine processes attributes and ontology and replies with a SAML AttributeStatement containing newly derived information.
5) The access control decision is computed using the loaded policies. It is now possible to use fine grained access control policies due to the expressiveness and completeness of available attribute information. The decision is sent back to the requesting entity, the IdP. The SAML authentication statement and the XACML authorization decision statement are referred back to the SP. Finally, a local PEP at the SP will enforce the decision.

VI. 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.

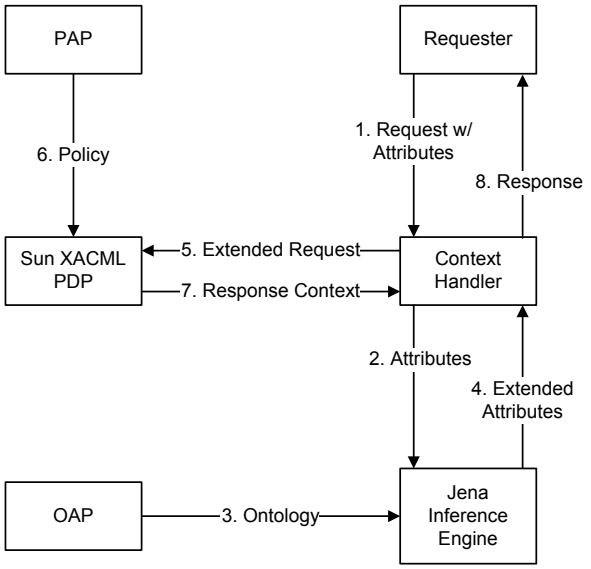

Figure 11. Architecture of the basic prototype implementation

The architecture of our prototype implementation is depicted in Fig. 11. The access control decision procedure is similar to section V. 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 chose Jena by Hewlett Packard[7] because it provides a framework for Semantic Web applications with persistent storage, RDF/OWL-support and SPARQL querying. Unfortunately, in the version we used (version 2.2) 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 (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[8] (version 1.2) as the PDP.

In parallel we have developed a prototype AAI based on the Liberty ID-FF 1.1 specification and integrated the Sun XACML reference implementation into the system. We are currently in the process of also integrating Jena and our extension as well in order to evaluate our approach in an AAI setting.

[7] http://jena.sourceforge.net
[8] http://sunxacml.sourceforge.net

VII. DISCUSSION

In the following, we discuss our approach in terms of performance, ontology and policy specification, and AAI aspects.

A. Performance

A performance evaluation of our semantic extension in comparison to a base XACML implementation was not done because of the multitude of other reasoners (e.g., RacerPro[9], Pellet[10], and Bossam[11]) available besides Jena. These reasoners possess a varying functional extent, e.g. Jena and Bossam have only partial reasoning support for the description logic variant of OWL, while Pellet and Racer have full support, yet only Pellet is sound and complete. On the other hand, Pellet and Jena do not support rules in SWRL while RacerPro and Bossam do, but only in fragments. It is because of these differences that a performance evaluation is not yet feasible. What is evident is that the time needed for an access control decision depends on the performance of base XACML implementation, the performance of deployed inference engine, and the complexity of ontology used, which with increasing complexity can lead to serious performance loss.

Naturally, the performance of the access control decision in an AAI heavily relies on the performance of our XACML extension. However, time and resources needed to collect attributes over the network from various service providers is far more costly.

B. Ontology and Policy Specification

At 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. These can be provided by corporate, national, or even international organizations, but also by software vendors 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 also expect that the widespread use of XACML will ensure the availability of powerful and easy-to-use policy editors.

However, there are some subtleties in ontology specification that have to be taken into account. One of the main issues lies in possible different interpretations of the knowledge base; one can reason on implicit knowledge encoded in an ontology according to the open world assumption (OWA) or the closed world assumption (CWA). These assumptions differ in their respective views of completeness of the information contained in a knowledge base. In OWA statements not existing in the knowledge base are considered as unknown, while in CWA they are considered as false. The OWA, which is standard to RDF/OWL reasoning, has advantages when dealing with

[9] http://www.racer-systems.com
[10] http://www.mindswap.org/2003/pellet
[11] http://projects.semwebcentral.org/projects/bossam

heterogeneous descriptions of identical concepts in open systems. Yet, problems with specification of concepts can arise. We refer the reader to [15] and [16] for an in-depth treatment and solution sketches as these issues are out of the scope of this paper.

C. AAI Aspects

We have shown a possible integration of our XACML extensions and the inference engine into an AAI. As AAI architectures differ tremendously the actual inclusion of the new components can differ likewise. The given approach is based on a distributed, decentralized federation. Here, the OAP and the inference engine were included into the access control decision point. The given architecture respects user privacy and separates identity information from profile information. The IdP is the only member in the federation actually knowing the user's identity. However, he is not aware of the resource accessed. The access control decision point is only aware of various attributes belonging to a pseudonym or any opaque identifier. Additional, inferred attributes are connected only with the pseudonym as well. The service provider is not presented with the true identity of the user. The request for attributes to a given identity is not connected with the original request.

VIII. Related Work

Comparative surveys on the functionality of existing AAIs can be found in [17] and [18]. Katsikas et all [19] sum up requirements in providing secure e-commerce.

As for employing Semantic Web technologies in the access control decision and enforcement process, an early approach has been presented by Yagüe et al. [20]. Their proposal defines its 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. A privilege management infrastructure (PMI) can be used to issue subjects' attribute certificates for use in access control; each issuing party produces semantic descriptions of the certificates it signs. Metadata of objects is described in a custom XML format. However, more powerful metadata representations like OWL and reasoning capabilities are not provided.

Damiani et al. [21] propose an semantically enhanced extension of XACML as well as a reference architecture for policy enforcement. They extend XACML 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. Damiani et al. use RDF for specification of metadata, thus not providing the richness of OWL. Furthermore, reasoning on subject metadata is not possible.

Rei [22] employs an ontology to represent concepts like rights and obligations relevant for a policy. Policies and rules can be defined in RDFS or a Prolog-like language. As our aim is to simplify attribute and policy management, we stick with the established policy language XACML and extend it by the use of OWL for attribute management.

Other approaches [23], [24], [25] use OWL instead of XACML as basis for the representation of policies. In [24] and [25] additionally SWRL is used to encode the knowledge base as well as the authorization rules.

IX. 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. This core 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. We then described how members of federations in AAIs can benefit from our facility and showed an integration into an AAI.

For future work, we plan to thoroughly evaluate the benefits of the presented approach. As mentioned before we are currently integrating our extensions in an AAI prototype and evaluating it in this context. Furthermore, we are in the process of developing a security infrastructure for an architecture based on semantic web services in the area of e-government. This is done in the context of the EU-funded project "Access-eGov"[12] which aims at building semantic interoperability among e-government services across organizational and regional borders. We also plan to provide ontologies for security interoperation which are appropriate for the presented approach.

References

[1] D.F. Ferraiolo, R. Sandhu, S. Gavrila, D. Kuhn, and R. Chandramouli, "Proposed NIST Standard for Role-based Access Control," *ACM Transactions on Information and Systems Security, Volume 4, Number 3*, August 2001, pp. 224-274.

[2] T. Priebe, E.B. Fernandez, J.I. Mehlau, and G. Pernul, "A Pattern System for Access Control," *Proc. 18th Annual IFIP WG 11.3 Working Conference on Data and Application Security*, Sitges, Spain, July 2004, pp. 235-249.

[3] T. Priebe, W. Dobmeier, B. Muschall, and G. Pernul, "ABAC – Ein Referenzmodell für attributbasierte Zugriffskontrolle," *Proc. 2. Jahrestagung Fachbereich Sicherheit der Gesellschaft für Informatik (Sicherheit 2005)*, Regensburg, Germany, April 2005, pp. 285-296.

[4] E. Yuan and J. Tong, "Attribute Based Access Control (ABAC) for Web Services," *Proc. 3rd International Conference on Web Services (ICWS 2005)*, Orlando, USA, July 2005, pp. 561-569.

[5] 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`

[6] T. Priebe, W. Dobmeier, N. Kamprath, "Supporting Attributed-based Access Control with Ontologies," *Proc. of the First International Conference on Availability, Reliability and Security (ARES 2006)*, Vienna, Austria, April 2006, pp. 465-472.

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

[12] http://www.access-egov.org

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

[9] C. Schläger, T. Nowey, and J.A. Montenegro, "A Reference Model for Authentication and Authorisation Infrastructures Respecting Privacy and Flexibility in b2c eCommerce," *Proc. of the First International Conference on Availability, Reliability and Security (ARES 2006)*, Vienna, Austria, April 2006, pp. 709-716.

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

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

[12] "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/

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

[14] C. Schlaeger, M. Sojer, B. Muschall, and G. Pernul, "Attribute-based Authentication and Authorisation Infrastructures for E-Commerce Providers," *Proc. of the 7th International Conference on Electronic Commerce and Web Technologies (EC-Web 2006)*, in press.

[15] E. Damiani, S. David, S. De Capitani di Vimercati, C.Fugazza, and P.Samarati, "Open World Reasoning in Semantics-Aware Access Control:a Preliminary Study," *Proc. 2nd Italian Semantic Web Workshop (SWAP 2005)*, Trento , Italy, December 2005.

[16] E. Damiani, S. De Capitani di Vimercati, C. Cristiano Fugazza, and P. Samarati, "Modality Conflicts in Semantics Aware Access Control," *Proc. 6th International Conference on Web Engineering (ICWE 2006)*, Menlo Park, USA, July 2006, pp. 249-256.

[17] J. Lopez, R. Oppliger, and G. Pernul, "Authentication and Authorization Infrastructures (AAIs): A Comparative Survey," *Computers & Security,* vol. 23, pp.578-590, 2004.

[18] C. Schläger and G. Pernul, "Authentication and Authorisation Infrastructures in b2c e-Commerce," *Proc. 6th International Conference on Electronic Commerce and Web Technologies (EC-Web 2005)*, Copenhagen, Denmark, September 2005, pp. 306-315.

[19] Sokratis K. Katsikas, Javier Lopez, and Günther Pernul, "Trust, Privacy and Security in E-business: Requirements and Solutions," *Proc. of the 10th Panhellenic Conference on Informatics (PCI'2005)*, Volos, Greece, November 2005, pp. 548-558.

[20] M. Yagüe, A. Mana, J. Lopez, and J.M. Troya, "Applying the Semantic Web Layers to Access Control," *Proc. of the DEXA 2003 Workshop on Web Semantics (WebS 2003)*, Prague, Czech Republic, September 2003, pp. 622-626.

[21] E. Damiani, S. De Capitani di Vimercati, C. Fugazza, and P. Samarati, "Extending Policy Languages to the Semantic Web," *Proc. Web Engineering - 4th International Conference (ICWE 2004)*, Munich, Germany, July 2004, pp. 330-343.

[22] L. Kagal, T. Finin, and A. Joshi, "A Policy Based Approach to Security for the Semantic Web," *Proc. 2nd International Semantic Web Conference (ISWC 2003)*, Sanibel Island, FL, October 2003, pp. 402-418.

[23] A. Uszok et al., "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, pp.93-98.

[24] H. Li, X. Zhang, H. Wu, and Y. Qu, "Design and Application of Rule Based Access Control Policies, " *Proc. Semantic Web and Policy Workshop in conjunction with the 4th International Semantic Web Conference (SWPW 2005)*, Galway, Ireland, November 2005. http://www.csee.umbc.edu/swpw/papers/zhang.pdf

[25] B. Shields, O. Molloy, G. Lyons, and J. Duggan, "Using Semantic Rules to Determine Access Control for Web Services," *Proc. 15th International Conference on World Wide Web (WWW 2006)*, Edinburgh, Scotland, May 2006, pp. 913-914.

Torsten Priebe holds a diploma degree in information systems from the University of Essen, Germany (2000) and a Ph.D. in economics from the University of Regensburg (2005).

He has been working for the Department of Information Systems at the University of Essen since 1999 and was involved in various international research projects of the group. Together with Prof. Günther Pernul he moved to the University of Regensburg in 2002. His main research interests are in the areas of data warehousing and business intelligence, knowledge management, and information security. In these areas he has published in international conference proceedings and journals and coauthored a textbook on data warehousing. Since 2006 he is working as a consultant for Capgemini, one of the world's leading providers of consulting, technology, and outsourcing services in Vienna, Austria.

Dr. Priebe is member of ACM, IEEE, AIS, and GI and has been serving on international program commitees, e.g., of the International Conference on Trust, Privacy, and Security in Digital Business (TrustBus), the International Conference on Information and Knowledge Management (CIKM), and the International Workshop on Web Semantics (WebS).

Wolfgang Dobmeier received a diploma degree in information systems from the University of Regensburg, Germany, in 2004.

He has been working as a software engineer for several small companies during his studies. In 2004, he joined the Department of Information Systems at the University of Regensburg as a research assistant and Ph.D. candidate in the group of Prof. Günther Pernul. His research interests include information and database security, federated systems, and their applications in e-government.

Mr. Dobmeier is a member of GI, the German informatics society.

Christian Schläger received a diploma degree in information systems from the University of Regensburg, Germany, in 2004.

He has been working as a software engineer and researcher for several small companies during and after his studies. In 2004, he joined the Department of Information Systems at the University of Regensburg as a research assistant and Ph.D. candidate in the group of Prof. Günther Pernul. Receiving a post-graduate DAAD grant he worked together with the group of Prof. Javier Lopez at the E.T.S. Ingenieria Informatica at the University of Málaga, Spain, doing research on security infrastructures. His research interests include information security and federated information systems as well as their applications in e-commerce.

Mr. Schläger is a member of GI, the German informatics society, ACM, and AIS.

Nora Kamprath holds a diploma degree in information systems from the University of Regensburg, Germany (2005).

She has been working at the Department of Bank Information Systems at the University of Regensburg and several banks and financial organizations during her studies. Since 2006 she is working as a consultant for KPMG in the Information Risk Management Department in Frankfurt/Main, Germany.

```
<?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 12. 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 13. Sample ontology in OWL/XML