REST Profile of XACML v3.0 Version 1.0
Committee Specification 02
23 November 2014

Specification URIs
This version:
http://docs.oasis-open.org/xacml/xacml-rest/v1.0/cs02/xacml-rest-v1.0-cs02.doc (Authoritative)
http://docs.oasis-open.org/xacml/xacml-rest/v1.0/cs02/xacml-rest-v1.0-cs02.html
http://docs.oasis-open.org/xacml/xacml-rest/v1.0/cs02/xacml-rest-v1.0-cs02.pdf

Previous version:
http://docs.oasis-open.org/xacml/xacml-rest/v1.0/cs01/xacml-rest-v1.0-cs01.doc (Authoritative)
http://docs.oasis-open.org/xacml/xacml-rest/v1.0/cs01/xacml-rest-v1.0-cs01.html
http://docs.oasis-open.org/xacml/xacml-rest/v1.0/cs01/xacml-rest-v1.0-cs01.pdf

Latest version:
http://docs.oasis-open.org/xacml/xacml-rest/v1.0/xacml-rest-v1.0.doc (Authoritative)
http://docs.oasis-open.org/xacml/xacml-rest/v1.0/xacml-rest-v1.0.html
http://docs.oasis-open.org/xacml/xacml-rest/v1.0/xacml-rest-v1.0.pdf

Technical Committee:
OASIS eXtensible Access Control Markup Language (XACML) TC

Chairs:
Hal Lockhart (hal.lockhart@oracle.com), Oracle
Bill Parducci (bill@parducci.net), Individual

Editor:
Rémon Sinnema (remon.sinnema@emc.com), EMC

Related work:
This specification is related to:

Abstract:
This specification defines a profile for the use of XACML in a RESTful architecture.

Status:
This document was last revised or approved by the OASIS eXtensible Access Control Markup Language (XACML) TC on the above date. The level of approval is also listed above. Check the “Latest version” location noted above for possible later revisions of this document. Any other numbered Versions and other technical work produced by the Technical Committee (TC) are listed at https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml#technical.

TC members should send comments on this specification to the TC’s email list. Others should send comments to the TC’s public comment list, after subscribing to it by following the instructions at the “Send A Comment” button on the TC’s web page at https://www.oasis-open.org/committees/xacml/.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the

**Citation format:**
When referencing this specification the following citation format should be used:

[XACML-REST-v1.0]

Notices

Copyright © OASIS Open 2014. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the “OASIS IPR Policy”). The full Policy may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS’ procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The name “OASIS” is a trademark of OASIS, the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see https://www.oasis-open.org/policies-guidelines/trademark for above guidance.
# Table of Contents

1  Introduction .......................................................................................................................... 5  
   1.1 Terminology ..................................................................................................................... 5  
   1.2 Glossary ............................................................................................................................ 5  
   1.3 Normative References ........................................................................................................ 5  
   1.4 Non-Normative References ............................................................................................... 6  
   1.5 Rationale ........................................................................................................................... 7  
      1.5.1 Externalization of Access Control ............................................................................. 7  
      1.5.2 Cloud Computing ....................................................................................................... 7  
      1.5.3 REST ............................................................................................................................ 8  
      1.5.4 RESTful Authorization as a Service .......................................................................... 8  
   1.6 Use Cases .......................................................................................................................... 9  
      1.6.1 PEP ↔ PDP ............................................................................................................... 9  
2  RESTful Services .................................................................................................................... 10  
   2.1 Network Transport .......................................................................................................... 10  
   2.2 Resources ......................................................................................................................... 10  
      2.2.1 Entry Point .................................................................................................................. 10  
      2.2.2 Policy Decision Point ............................................................................................... 10  
   2.3 Representations ................................................................................................................. 11  
      2.3.1 Linking ......................................................................................................................... 11  
      2.3.2 Entry Point .................................................................................................................. 12  
      2.3.3 XACML versions, Representation Formats, and Content Negotiation ..................... 12  
   2.4 Examples .......................................................................................................................... 12  
      2.4.1 Obtain an Access Decision ....................................................................................... 12  
3  Security Considerations ......................................................................................................... 14  
   3.1 Network Transport .......................................................................................................... 14  
   3.2 Authentication .................................................................................................................. 14  
   3.3 Authorization .................................................................................................................... 14  
   3.4 Non-Repudiation .............................................................................................................. 14  
4  Conformance .......................................................................................................................... 15  
   4.1 Conformance Clauses ....................................................................................................... 15  
   4.2 Test Assertions .................................................................................................................. 15  
      4.2.1 Network Transport ..................................................................................................... 15  
      4.2.2 Entry Point .................................................................................................................. 15  
      4.2.3 Policy Decision Point ............................................................................................... 16  
Appendix A. Acknowledgments ................................................................................................. 18  
Appendix B. Revision History .................................................................................................... 19
1 Introduction

{Non-normative}
This specification defines a profile for the use of the OASIS eXtensible Access Control Markup Language (XACML), versions 3.0 [XACMLv3] and earlier. Use of this profile requires no changes or extensions to the XACML standard.

This specification assumes the reader is somewhat familiar with XACML. A brief overview of XACML is available in [XACMLIntro].

This specification begins with a discussion of the topics and terms of interest in this profile. It then describes the details of RESTful services that conforming implementations must support. All sections of this profile are normative unless explicitly stated otherwise.

1.1 Terminology
The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

1.2 Glossary

Client
The agent that initiates requests to a server.

Representation
A sequence of bytes, in a given format, that represents a resource in some way.

Request
The HTTP request message sent from the client to the server [HTTPMessage]. Note that this is not the same concept as a XACML request [XACMLv3].

Resource
A service that is offered by the server [REST]. This can be static, like a document, or dynamic, like a search. Note that this is not the same concept as a XACML resource [XACMLv3].

Response
The HTTP response message returned from the server to the client [HTTPMessage]. Note that this is not the same concept as a XACML response [XACMLv3].

Server
The agent that handles requests from a client.

1.3 Normative References


1.4 Non-Normative References


http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf


http://www.iana.org/assignments/media-types/application/vnd.siren+json

https://github.com/mamund/media-types/blob/master/uber-hypermedia.asciidoc

[XACMLIntro]  A Brief Introduction to XACML. 14 March 2003,  
http://www.oasis-open.org/committees/download.php/2713/Brief_Introduction_to_XACML.html

1.5 Rationale

{Non-normative}

1.5.1 Externalization of Access Control

XACML [XACMLIntro] can be used for controlling access within a single application. This removes hard-coded security constraints from the application code, making it easier to change them. It also makes it possible to use a standard Policy Decision Point (PDP), so that organizations can make a proper make-or-buy decision. For virtually all organizations, authorization is not their core business, so being able to use an off-the-shelf product is appealing.

Although these are substantial benefits, XACML really shines when authorization is completely externalized from the application. Policies can then be reused across many applications, each using the same PDP. This leads to greater consistency of access control rules and improved efficiency in maintaining them.

1.5.2 Cloud Computing

Once access control policies are externalized from the application, the PDP can become a service to be shared in a cloud computing scenario.

The National Institute of Standards and Technology (NIST) defines cloud computing as “a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.” [Cloud].
Applying the ideas of cloud computing to access control leads to Authorization as a Service (AZaaS). The Cloud Security Alliance sees this as part of the Identity and Access Management category of service that they distinguish in the Security as a Service field [SecaaS]. Note that AZaaS requires a much heavier load on servers than Authentication as a Service, since authentication happens only once for a user session, while authorization must occur on every user action.

### 1.5.3 REST

In cloud computing, services are shared and must therefore be accessed over a computer network. Cloud infrastructure will thus by definition have a network-addressable API. Such an API can be built on RESTful principles.

**REpresentational State Transfer** (REST) is a system of architectural constraints that govern the interaction between a client and a server [REST]. In cloud computing, the client is the cloud service consumer, and the server is the cloud service itself. The constraints that REST adds to a client-server system are:

1. **Statelessness**: Each request from client to server must contain all of the information necessary to understand the request, and cannot take advantage of any stored context on the server. It improves visibility, reliability and scalability.
2. **Cache**: Data within a response to a request must be implicitly or explicitly labeled as cacheable or non-cacheable. It improves efficiency and scalability.
3. **Uniform interface**: Client and server interact through a generalized interface. It improves visibility, simplicity and evolvability, at the expense of efficiency. This is the distinguishing feature of REST. The constraints on the generalized interface are:
   i. **Identification of resources**: The key abstraction of information in REST is a resource. Any information that can be named can be a resource: a document or image, a temporal service, a collection of other resources, a non-virtual object, and so on. Each resource is identified by a resource identifier. In practice, this will be a Uniform Resource Identifier [URI].
   ii. **Manipulation of resources through representations**: Actions on resources are performed on representations of those resources. A representation is a sequence of bytes, plus representation metadata to describe those bytes. In practice, representations will be described by MIME media types [Media].
   iii. **Self-descriptive messages**: All the information required to process a request is available in the request. This includes the host, message control metadata (like Content-Length), representation metadata and the resource representation.
   iv. **Hypermedia as the engine of application state (HATEOAS)**: The client knows only the starting URL of the server. All future interactions are discovered from representations. This allows the server to evolve separately from the clients.
4. **Layered system**: Clients and servers can be composed of hierarchical layers such that each component cannot see beyond the immediate layer with which it is interacting. It improves simplicity and scalability at the expense of efficiency.
5. **Code-on-demand**: Client functionality can be extended by downloading and executing code in the form of applets or scripts. It improves simplicity and extensibility at the expense of visibility and security. This is an optional constraint.

The constraints of a RESTful architecture lead to simple, scalable, and evolvable systems. Simplicity means that few demands are placed on the cloud service consumer, whereas scalability and evolvability let the cloud service meet its rapid provisioning and releasing requirements, while incrementally expanding its services.

### 1.5.4 RESTful Authorization as a Service

Due to the pervasive nature of access control, Authorization-as-a-Service will result in many calls to the authorization servers. These servers must therefore perform and scale extremely well. Thus it makes sense to use a RESTful architecture for them.
This specification defines a profile for the use of XACML in a RESTful architecture, enabling the interoperability of RESTful Authorization-as-a-Service (AZaaS) solutions. The MIME media types [Media] available for representations of the various XACML constructs are defined separately [XACMLMedia].

1.6 Use Cases

This version of this profile will only consider the PEP and PDP. Later versions may involve other components of the XACML architecture, like the PAP and PIP.

1.6.1 PEP ↔ PDP

Line Of Business applications contain Policy Enforcement Points (PEPs) that interact with Policy Decision Points (PDPs) from various vendors. These PDPs may either be dedicated to the application, or be simultaneously offered to multiple applications (Authorization as a Service).
2 RESTful Services

2.1 Network Transport

The following URI SHALL be used as the identifier for the functionality specified in this section of this profile:

- urn:oasis:names:tc:xacml:3.0:profile:rest:http

Although not strictly required by REST, this specification mandates that HTTP MUST be used as the protocol to transport network messages [HTTPMessage] between client and server. For additional security, it is RECOMMENDED that SSL/TLS be used [HTTPS]. See section 3, Security Considerations, for more on securing the RESTful interactions. Note that additional transport protocols are allowed but outside the scope of this profile.

2.2 Resources

The following sections describe the mandatory and optional resources that this profile defines. Each section defines which operations are supported on the resource, and what their requirements are. In particular, HTTP status codes [HTTPSemantics] define success or failure of the operation. See section 3, Security Considerations, for information on securing the RESTful interactions and representations.

2.2.1 Entry Point

The following URI SHALL be used as the identifier for the functionality specified in this section of this profile:

- urn:oasis:names:tc:xacml:3.0:profile:rest:home

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request Body</th>
<th>Response Body</th>
<th>Description</th>
<th>Status Codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td></td>
<td>XACML entry point</td>
<td></td>
<td>200, 400, 401, 403, 406, 5xx</td>
</tr>
</tbody>
</table>

To enable the discoverability requirement, a RESTful XACML system MUST have a single entry point at a known location (the “billboard URI”). It is RECOMMENDED that the location of the entry point remain fixed, even as the service evolves, to allow older clients to remain functional. Each implementation of this profile MUST document the location of the entry point.

Note that the XACML entry point MAY be part of a larger RESTful system. In that case, the entry point location is not known in advance, but discovered from the enclosing system. The link relation http://docs.oasis-open.org/ns/xacml/relation/home SHALL be used for links to this resource. The documentation SHOULD contain information on how to discover the XACML entry point using this link relation.

The XACML entry point representation that is returned SHOULD NOT contain anything other than links to other resources specified in this profile.

2.2.2 Policy Decision Point

The following URI SHALL be used as the identifier for the functionality specified in this section of this profile:

- urn:oasis:names:tc:xacml:3.0:profile:rest:pdp

The link relation for links to this resource is http://docs.oasis-open.org/ns/xacml/relation/pdp.
A **server** MUST support `<Request>` from XACML core [XACMLv3] as the XACML request in the request body.

A **server** MAY additionally support `<XACMLAuthzDecisionQuery>` from the SAML Profile [SAML4XACML] as the XACML request. When `<XACMLAuthzDecisionQuery>` is used, requests and responses can be correlated using the request's ID and the response's InResponseTo attributes. When `<Request>` is used, this additional functionality is not available and the PEP must either use a new TCP/IP session, or wait with sending a request over the current session until the response for the previous request is received.

The processing and response MUST be as specified in the respective specification, either [XACMLv3] or [SAML4XACML].

The **POST** method is used rather than **GET** because the XACML request may contain sensitive data and because of practical considerations like limits on URI length [HTTPMethod]. Although the **POST** method is used, this operation is both idempotent and safe [HTTPSemantics]. It is **RECOMMENDED** that **servers** include cache control headers [HTTPCache] in their **responses** to make this explicit.

Note that success of the HTTP operation (i.e. status code 200) doesn’t mean that authorization is granted. It means that the **response** body is valid, and that the **response** body contains the XACML decision, which could be Deny. Likewise, a status code of 403 doesn’t imply a XACML decision of Deny, but instead means that the user is not allowed to ask the PDP for an access decision.

### 2.3 Representations

XACML requests and responses **SHOULD** be represented using registered media types defined specifically for XACML, in [XACMLMedia] or elsewhere (e.g. in the upcoming JSON profile).

Other representations **MAY** use any media type that matches the constraints outlined in the remainder of this section.

#### 2.3.1 Linking

A fundamental concept in a RESTful architecture is that of linking between **resources** [REST]. It is therefore of the utmost importance to use media types, like the Atom Syndication Format [Atom], that define linking structures. Such media types are also referred to as hypermedia types.

There has been a spur of activity in this space recently. Interesting examples are [Mason], [Siren], and [UBER].

The link relation types [WebLink] for the services in this specification are given in their respective sections below. For instance, a link to the PDP in Atom [Atom] would be represented as

```xml
<atom:link rel="http://docs.oasis-open.org/ns/xacml/relation/pdp" href="/authorization/pdp"/>
```

The same link to the PDP in Siren [Siren] would be represented as

```json
"links": [{
  "rel": "http://docs.oasis-open.org/ns/xacml/relation/pdp",
  "href": "/authorization/pdp"
}]
```

Whenever it is not possible to add links to a **representation**, for instance because the **representation** must conform to a schema that doesn’t support links, or because the **representation** is binary, links...
MUST be added using the Link HTTP header [WebLink]. In case of multiple links, the title attribute of the Link header field MAY be used to correlate the link to an item in the representation.

2.3.2 Entry Point

The representation of the entry point resource SHOULD NOT contain anything other than links to other resources specified by this profile.

2.3.3 XACML versions, Representation Formats, and Content Negotiation

This profile is agnostic to the version and format of XACML used. Clients and servers SHOULD use content negotiation [HTTPSemantics] to agree on the version and format of XACML used in their exchanges. XACML media types[Media] MUST support a version parameter for this purpose. See the Examples section for the use of the Accept and Content-Type headers [HTTPSemantics] used in content negotiation.

It's an error for clients or servers to send a body where the representation doesn't match the Content-Type. Servers MUST return 400 Bad Request when a client does so.

HTTP allows Content-Type and Accept to be different, so a client could supply a JSON-formatted request and ask for an XML-formatted response or vice versa. Although this profile doesn't define a use case for that, it also doesn't forbid it. Servers that are not willing to honor such requests SHOULD either return 406 Not Acceptable or simply return the response in the format of their choosing.

2.4 Examples

(Non-normative)

2.4.1 Obtain an Access Decision

The following is an example sequence of HTTP messages to obtain an authorization decision from a PDP. The client starts by accessing the entry point:

```
GET /authorization HTTP/1.0
Host: www.example.com
Accept: application/xml
Content-Length: 0
```

To which the server responds:

```
HTTP/1.0 200 OK
Content-Type: application/home+xml
Content-Length: <nnnn>

<?xml version="1.0"?>
<resources xmlns="http://ietf.org/ns/home-documents" >
 <resource rel="http://docs.oasis-open.org/ns/xacml/relation/pdp">
 <link href="/authorization/pdp/"/>
 </resource>
</resources>
```

In this example, the response follows the XML syntax for home documents [HomeDocXml].
The **client** looks for a resource with relation type http://docs.oasis-open.org/ns/xacml/relation/pdp and **POSTs** the XACML request to it:

```xml
POST /authorization/pdp/ HTTP/1.0
Host: www.example.com
Accept: application/xacml+xml; version=3.0
Content-Type: application/xacml+xml; version=3.0
Content-Length: <nnnn>

<?xml version="1.0"?><Request xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"><!-- XACML request --></Request>
```

And finally the **server** responds with the access decision:

```xml
HTTP/1.0 200 OK
Content-Type: application/xacml+xml; version=3.0
Content-Length: <nnnn>

```
3 Security Considerations

Security and privacy considerations for the use of XACML in general are defined in [XACMLv3]. This section describes some additional considerations that have to do with the networked nature of a RESTful architecture, together with the administrative capabilities set out by this profile.

3.1 Network Transport

The use of SSL/TLS [HTTPS] is RECOMMENDED to protect data as it is transferred across the network.

3.2 Authentication

This specification leaves the issue open of how to authenticate the requestor. Implementations MUST document how they handle authentication.

HTTP status code 401 (Unauthorized) [HTTPAuthN] MAY be used to indicate that an operation on a resource is denied because the requestor is not authenticated. However, the problem of authentication over HTTP is not completely solved. [HTTPAuthN] defines Basic and Digest authentication. Basic authentication MUST NOT be used, since it sends the password in plain text over the network. Digest authentication MAY be used.

Additional standards like [OpenID], [OAuth], [SAMLv2], or [SASL] MAY be used instead of or in addition to HTTP Digest authentication.

3.3 Authorization

This specification RECOMMENDS that authorization be implemented using XACML. Implementations can perform authorization based upon the identity of the requestor, as well as on any appropriate additional, trusted, attribute. The use of the XACML Administration and Delegation Profile [Admin] is RECOMMENDED.

HTTP status code 403 (Forbidden) [HTTPSemantics] MUST be used to indicate that an operation on a resource is denied because the requestor is not authorized.

Authorization SHOULD be used to exclude from the response any links to resources that the requestor is not allowed to access.

3.4 Non-Repudiation

In some situations it is important to have an audit trail of access decisions that were made. This audit trail must be at least tamper-evident. For this purpose, the SAML Profile for XACML [SAML4XACML] can be used to sign the access request and response.
4 Conformance

{Normative}

4.1 Conformance Clauses

This section lists those portions of the specification that MUST be included in an implementation of a server that claims to conform to this profile.

<table>
<thead>
<tr>
<th>Identifier</th>
<th>M/O</th>
</tr>
</thead>
<tbody>
<tr>
<td>urn:oasis:names:tc:xacml:3.0:profile:rest:http</td>
<td>M</td>
</tr>
<tr>
<td>urn:oasis:names:tc:xacml:3.0:profile:rest:home</td>
<td>M</td>
</tr>
<tr>
<td>urn:oasis:names:tc:xacml:3.0:profile:rest:pdp</td>
<td>M</td>
</tr>
</tbody>
</table>

4.2 Test Assertions

This section lists test assertions [TAG] that help verify conformance to this specification.

4.2.1 Network Transport

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Normative Source</td>
<td>The client must use HTTP when communicating with the server (From the more general source: HTTP MUST be used as the protocol to transport network messages between client and server)</td>
</tr>
<tr>
<td>Target</td>
<td>Network message from the client</td>
</tr>
<tr>
<td>Predicate</td>
<td>The [message] starts with an HTTP request line [HTTPMessage]</td>
</tr>
<tr>
<td>Prescription Level</td>
<td>mandatory</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Normative Source</td>
<td>The server must use HTTP when communicating with the client (From the more general source: HTTP MUST be used as the protocol to transport network messages between client and server)</td>
</tr>
<tr>
<td>Target</td>
<td>Network message from the server</td>
</tr>
<tr>
<td>Predicate</td>
<td>The [message] starts with an HTTP response line [HTTPMessage]</td>
</tr>
<tr>
<td>Prescription Level</td>
<td>Mandatory</td>
</tr>
</tbody>
</table>

4.2.2 Entry Point

<table>
<thead>
<tr>
<th>Id</th>
<th>urn:oasis:names:tc:xacml:3.0:profile:rest:assertion:home:documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normative Source</td>
<td>A RESTful XACML system MUST have a single entry point at a known location. Each implementation of this profile MUST document the location of the entry point</td>
</tr>
<tr>
<td>Target</td>
<td>server documentation</td>
</tr>
<tr>
<td>Predicate</td>
<td>The [documentation] lists a (procedure for discovering a) single entry point URL at which the server can be accessed</td>
</tr>
</tbody>
</table>
### Prescription Level

<table>
<thead>
<tr>
<th>Id</th>
<th>urn:oasis:names:tc:xacml:3.0:profile:rest:assertion:home:status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normative Source</td>
<td>GET on the entry point location MUST return status code 200</td>
</tr>
<tr>
<td>Target</td>
<td>Response to GET request on the entry point location</td>
</tr>
<tr>
<td>Predicate</td>
<td>The HTTP status code in the [response] is 200</td>
</tr>
<tr>
<td>Prescription Level</td>
<td>Mandatory</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Id</th>
<th>urn:oasis:names:tc:xacml:3.0:profile:rest:assertion:home:pdp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normative Source</td>
<td>The XACML entry point representation SHOULD contain a link to the PDP</td>
</tr>
<tr>
<td>Target</td>
<td>Response to GET request on the entry point location</td>
</tr>
<tr>
<td>Predicate</td>
<td>The [response] body contains a resource with link relation <a href="http://docs.oasis-open.org/ns/xacml/relation/pdp">http://docs.oasis-open.org/ns/xacml/relation/pdp</a> and a valid URL</td>
</tr>
<tr>
<td>Prescription Level</td>
<td>mandatory</td>
</tr>
</tbody>
</table>

### 4.2.3 Policy Decision Point

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Normative Source</td>
<td>POST on the PDP with a valid XACML request MUST return status code 200</td>
</tr>
<tr>
<td>Target</td>
<td>Response to POST request on the PDP location with valid XACML request in the body</td>
</tr>
<tr>
<td>Predicate</td>
<td>The HTTP status code in the [response] is 200</td>
</tr>
<tr>
<td>Prescription Level</td>
<td>mandatory</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Normative Source</td>
<td>POST on the PDP with a valid XACML request MUST return a valid XACML response in the body</td>
</tr>
<tr>
<td>Target</td>
<td>Response to POST request on the PDP location with valid XACML request in the body</td>
</tr>
<tr>
<td>Predicate</td>
<td>The HTTP body in the [response] is a valid XACML response</td>
</tr>
<tr>
<td>Prescription Level</td>
<td>mandatory</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Normative Source</td>
<td>POST on the PDP with an invalid XACML request MUST return status code 400 (Bad Request)</td>
</tr>
<tr>
<td>Target</td>
<td>Response to POST request on the PDP location with invalid XACML request in the body</td>
</tr>
<tr>
<td>Predicate</td>
<td>The HTTP status code in the [response] is 400</td>
</tr>
<tr>
<td>Prescription Level</td>
<td>mandatory</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Normative Source</td>
<td>POST on the PDP with a valid XACML request MUST return status code 200</td>
</tr>
<tr>
<td>Target</td>
<td>Response to <strong>POST request</strong> on the PDP location with valid XACML request wrapped in a xacml-samlp:XACMLAuthzDecisionQuery in the body</td>
</tr>
<tr>
<td>--------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Predicate</td>
<td>The HTTP status code in the [response] is 200</td>
</tr>
<tr>
<td>Prescription Level</td>
<td>optional</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Normative Source</td>
<td><strong>POST</strong> on the PDP with a valid XACML request <strong>MUST</strong> return a valid XACML response in the body</td>
</tr>
<tr>
<td>Target</td>
<td>Response to <strong>POST request</strong> on the PDP location with valid XACML request wrapped in a xacml-samlp:XACMLAuthzDecisionQuery in the body</td>
</tr>
<tr>
<td>Predicate</td>
<td>The HTTP body in the [response] is a valid XACML response wrapped in a samlp:Response</td>
</tr>
<tr>
<td>Prescription Level</td>
<td>optional</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Normative Source</td>
<td><strong>POST</strong> on the PDP with an invalid XACML request <strong>MUST</strong> return status code 400 (Bad Request)</td>
</tr>
<tr>
<td>Target</td>
<td>Response to <strong>POST request</strong> on the PDP location with invalid XACML request wrapped in a xacml-samlp:XACMLAuthzDecisionQuery in the body</td>
</tr>
<tr>
<td>Predicate</td>
<td>The HTTP status code in the [response] is 400</td>
</tr>
<tr>
<td>Prescription Level</td>
<td>optional</td>
</tr>
</tbody>
</table>
Appendix A. Acknowledgments

The following individuals have participated in the creation of this specification and are gratefully acknowledged:

**Participants:**
- David Brossard, Axiomatics
- Jean-Paul Buu-Sao, TSCP
- David Chadwick, Individual
- Jacques Durand, Fujitsu
- Craig Forster, IBM
- Hal Lockhart, Oracle
- Danny Thorpe, Dell
- Erik Wilde, EMC
## Appendix B. Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Editor</th>
<th>Changes Made</th>
</tr>
</thead>
<tbody>
<tr>
<td>WD01</td>
<td>2012-02-14</td>
<td>Rémon Sinnema</td>
<td>Defined use cases</td>
</tr>
<tr>
<td>WD02</td>
<td>2012-04-24</td>
<td>Rémon Sinnema</td>
<td>Initial full draft</td>
</tr>
<tr>
<td>WD03</td>
<td>2012-05-03</td>
<td>Rémon Sinnema</td>
<td>Fixed typos&lt;br&gt;Renamed Use Cases section to Rationale&lt;br&gt;Introduced Use Cases section&lt;br&gt;Move everything representation related out of the section on resources&lt;br&gt;Added examples&lt;br&gt;Improved authorization section</td>
</tr>
<tr>
<td>WD04</td>
<td>2012-05-22</td>
<td>Rémon Sinnema</td>
<td>Conformance section should succinctly indicate what needs to be implemented&lt;br&gt;Added platform use case&lt;br&gt;Added policy version resource</td>
</tr>
<tr>
<td>WD05</td>
<td>2012-05-31</td>
<td>Rémon Sinnema</td>
<td>PDP is now optional, allowing PAP-only servers&lt;br&gt;Added explanatory text for delete example&lt;br&gt;Added note on policies contained within policy sets&lt;br&gt;Added note that supplied policies must be valid according to the policy schema&lt;br&gt;Improved wording in Security section&lt;br&gt;Added “lost” paragraph from WD02 about the contents of the entry point resource&lt;br&gt;Added text on different types of PAPs&lt;br&gt;Added text on policy (version) equality&lt;br&gt;Added use of HTTP to conformance section</td>
</tr>
<tr>
<td>WD06</td>
<td>2012-10-9</td>
<td>Rémon Sinnema</td>
<td>Added domain terms&lt;br&gt;Added section on test assertions&lt;br&gt;Removed policy administration related text&lt;br&gt;Updated text to better fit the home document standard&lt;br&gt;Added section on non-repudiation&lt;br&gt;Replaced reference to XACML Media Types Profile with URL of Internet Draft&lt;br&gt;Added text on embedding XACML REST in a larger RESTful system</td>
</tr>
<tr>
<td>WD07</td>
<td>2013-01-25</td>
<td>Rémon Sinnema</td>
<td>Added section on content negotiation</td>
</tr>
<tr>
<td>WD08</td>
<td>2014-06-10</td>
<td>Rémon Sinnema</td>
<td>Renamed to follow profile naming convention.&lt;br&gt;Fixed normative/non-normative indicators on sections.&lt;br&gt;Updated HTTP RFCs.&lt;br&gt;Added text on HTTP level caching.&lt;br&gt;Added explanation on the use of POST rather than GET.&lt;br&gt;Replaced XACML media type draft with RFC.&lt;br&gt;Added text on hypermedia types.&lt;br&gt;Added text on sending JSON requests and asking for XML responses or vice versa.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Relaxed constraints on entry point representation.</td>
<td></td>
</tr>
</tbody>
</table>