

SAML–XACML Authorisation Interface and XACML Obligations Handling

EGEE JRA1 Technical Document, Draft version 0.3, April 29, 2008

Editor - Yuri Demchenko

Contributors – Yuri Demchenko, Oscar Koeroo, Hakon Sagehaug, Valerio Venturi

Goal and scope of this document

The goal of this document is to provide an basic information and initial design suggestions for building interoperable SAML-XACML Authorisation Interface based on the OASIS SAML2.0 profile of XACML specification [1] that can be used as a component of the distributed Authorisation infrastructure for Grid services and applications.

SAML2.0 profile of XACML combines well established SAML security assertions format [2] and reach functionality of the XACML policy format [3]. The SAML-XACML interface is recommended as the standard protocol for Grid oriented AuthZ services by OGF OGSA AUTHZ-WG [4]. SAML-XACML interface is also recommended as a common interface for Grid resource AuthZ services and Site Central AuthZ Service by EGEE-OSG AuthZ Interoperability workshop [5]

The document summarises the recent and on-going discussions among cooperative Grid infrastructure projects such as EGEE, OSG and middleware development projects and initiatives such as gLite, Globus and others. This activity is also considered in relation/connection to the Authorisation standardisation initiative/activity by the OGSA AUTHZ-WG at the OGF.

The document provides a short overview of the Obligations definition in the XACML specification and provides examples of the XACML obligations expression for the basic use cases in Grid Authorisation service that require user/requestor ID mapping from the Grid ID to the local Unix ID that can be managed as a pool-accounts set. Using XACML for modeling Grid AuthZ policies provides a reference framework for ensuring interoperability between both XACML and non-XACML based AuthZ services.

This study is conducted in the context of the general JRA1 AuthZ practice study and with the reference to ongoing development of the Site Central AuthZ Service (SCAS) for Grid application and resources. Site-central AuthZ service means that all site located resources and services use central AuthZ service that maintains a common set of policies for this site.

This is not in the scope of this document to provide a detailed description of the currently tested implementations of the SAML-XACML on-wire libraries being developed in the framework of different projects and collaborations such as by Globus, EGEE and CNAF. Instead the document is focused on the common operational models and conformance tests to ensure their interoperability.

<p>Note: This document doesn't go into possible inter-project coordination and individual projects management issues, however in some places it refers to existing/developing solutions and implementations. More details on the existing implementations will be provided in the later versions of the document.</p>

Content

1 Introduction	2
2 Authorisation service implementation in EGEE and OSG and basic use cases	3
2.1 Current components of the AuthZ service infrastructure	3
2.2 Obligations enforcement scenarios	4
2.3 Policy Obligations and their use in the typical Grid oriented authorisation scenarios	5
3 Obligations processing model and suggested implementation	5
3.1 Generic Authorisation Request processing flow and Obligations Handling Reference Model (OHRM)	5
3.2 Authorisation Request processing flow and Obligations handling model in gJAF	7
4 Obligations expression in the XACML policy and proposed conventions	8
4.1 Obligations definition and expression in the XACML policy	8
4.2 Examples of expressing and handling Obligations	9
4.3 Proposed Obligation expression conventions	15
5 Obligation Handling API	16
6 Conformance test	16

1 Introduction

The document starts from the short overview of the current AuthZ service implementations in the Grid applications in the two major Grid infrastructure projects EGEE and OSG and based on two major middleware framework gLite and Globus. The goal of this overview (which doesn't pretend for completeness in any way) is to define the basic use cases for using common SAML-XACML interface and common Obligations handling model.

In the following section, the document proposes the general obligations handling model referred to as the Obligations Handling Reference Model (OHRM). This is followed by the Obligations expression examples and suggestions regarding ObligationId format

As a supplementary material, the document provides examples of the XACML policies that contain suggested/registered Obligations that will constitute in the future the compliance test.

The document also contains reference and illustrative material from the XACML2.0, SAML2.0 specifications and the SAML2.0 profile of XACML specification which are placed at the end of document and in Appendices.

Summary of the proposed design suggestions and solutions for the SAML-XACML interface implementation:

- 1) Two basic use cases of the possible SCAS implementation – LCAS/LCMAPS based and native XACML based, that correspondently implement stateful and stateless PDP operational model.
- 2) Description of different obligation enforcement scenarios, including obligations enforcement at run-time at the moment when the obligated AuthZ decision received by the PEP, and delayed at later time when a user/requestor accesses the reserved service.
- 3) Obligations Handling Reference Model (OHRM).
- 4) (Conventional) agreement on the Obligations expression in the XACML policy and applicable XACML Request format.
- 5) ObligationId format and OHRM related Obligation marking/labelling approach.
- 6) Basic (design) requirements to the ObligationHandler API.
- 7) SAML2.0-XACML profile conformance test definition and requirements to ensure interoperability between different profile implementations against the standard and proposed Obligations handling conventions.

2 Authorisation service implementation in EGEE and OSG and basic use cases

2.1 Current components of the AuthZ service infrastructure

Current implementation/organisation of the AuthZ infrastructure in EGEE and gLite based Grids, on one side, and OSG and Globus based Grids, on other side, uses different but interoperable components and models, but interoperability is achieved by using implementation specific gateways and not by the common implementation independent interface use.

Figure 2.1 below illustrate current architecture. The typical Grid resource setup uses LCAS/LCMAPS AuthZ service that is installed per Grid resource or Worker node (WN) but often uses the same site or resource specific policy in the form of grid/groupmapfiles. It is important to mention in the context of current study that LCAS/LCMAPS service requires resource or node specific configuration file/information. The following issues have been identified to ensure access control consistency among number of Grid resources and different middleware platform:

- share/distribute the gridmapdir for mapping consistency
- share/distribute the configurations for the nodes
- share/distribute authorization files, like grid/groupmapfiles and a blacklisting file
- scaling issues; lots of node will probably overload an NFS server

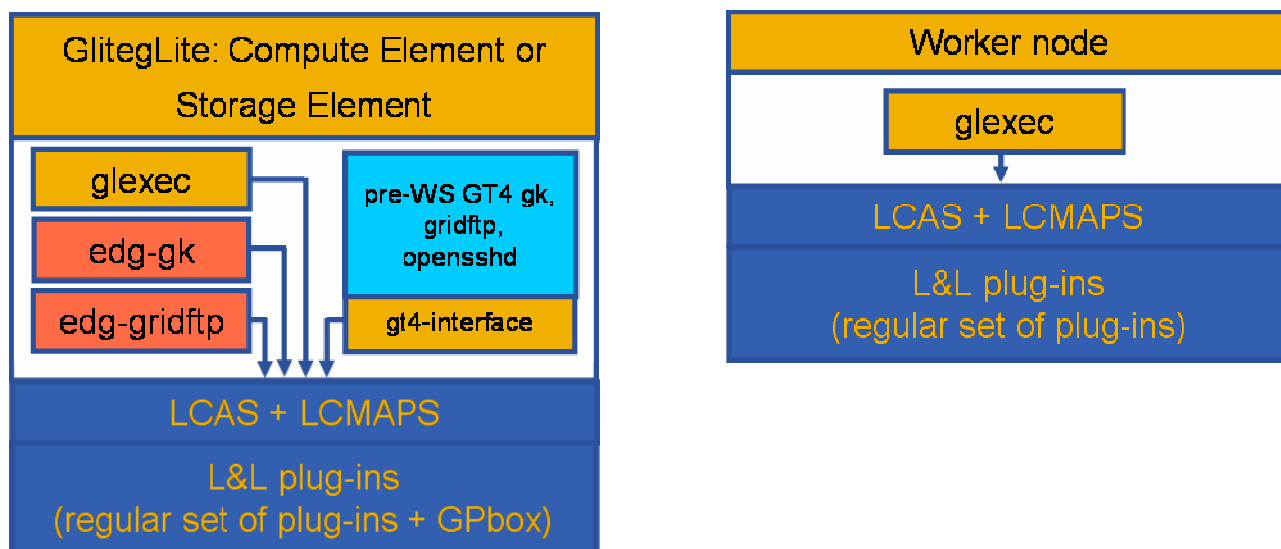


Fig. 2.1. Current AuthZ services architectural components

Figure 2.2. illustrates proposed architecture that intend to address mentioned above issues by introducing Site Central AuthZ Service (SCAS). The SCAS can use different AuthZ service implementations such as LCAS/LCAMAPS, gPlazma/Prima, or G-PBox that use both non-XACML and XACML policy formats.

The proposed architecture/model/solution should provide interoperability for both types of AuthZ service – XACML and not XACML. Interoperability between different SCAS implementations and Grid resources AuthZ services is achieved by introducing a common SAML-XACML interface layer that implements SAML2.0 profile of XACML and provides both SAML-XACML wire communication protocol and adjusting AuthZ service specific policy and Request/Response semantics to the semantics and format of the SAML2.0 profile of XACML.

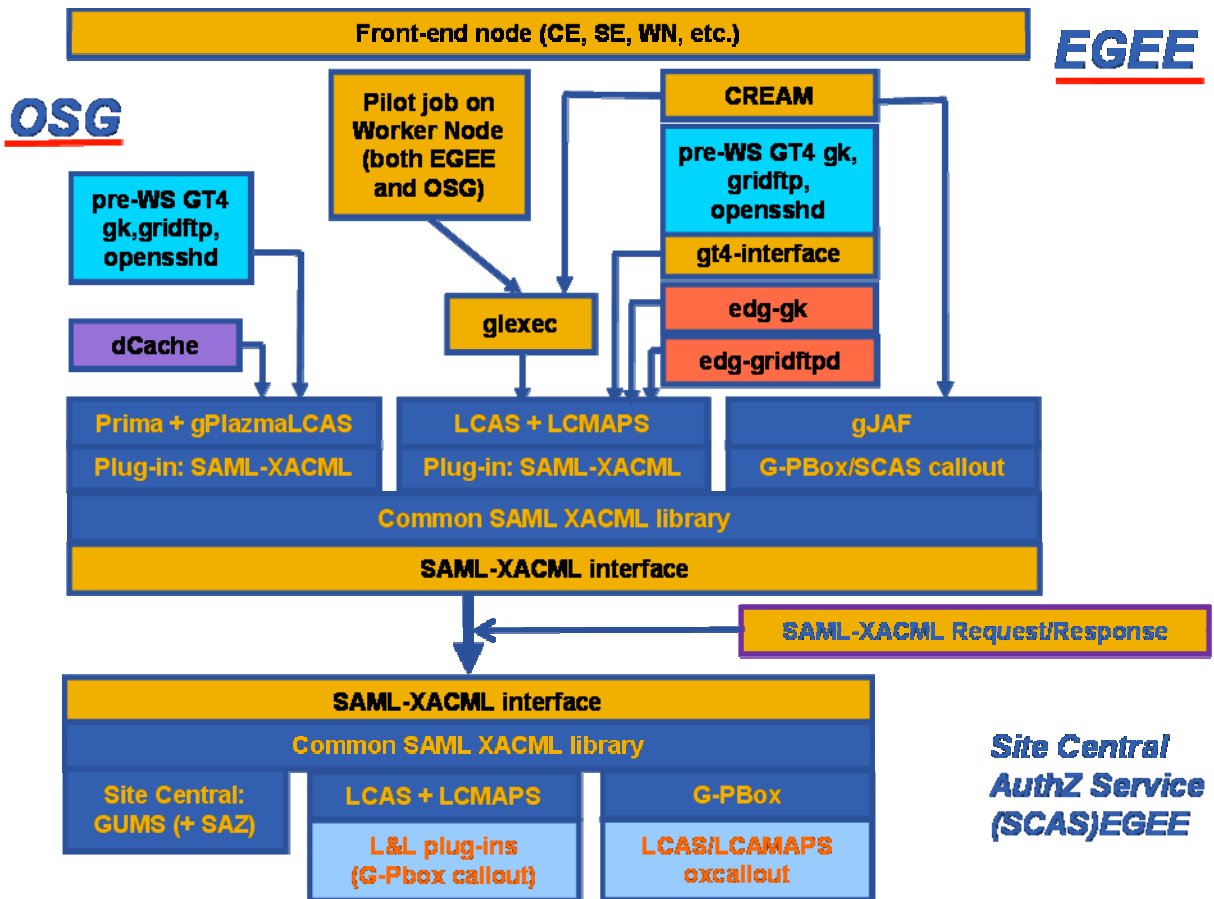


Fig. 2.2. Suggested interoperability mesh using common SAML-XACML library

For the analysis and design purposes, the whole policy based AuthZ process can be split on two stages – policy decision and policy enforcement which are correspondently supported by such functional components as PDP and PEP which are typically functionally separated. Policy decision can be outsourced to the external to the resource AuthZ service like in case with SCAS. Policy enforcement is typically executed under the resource control/privileges.

Given typical stateless AuthZ policy definition as a set of attribute based access rules the PDP supporting such a policy can be also considered stateless.

In the particular case of the LCAS/LCMAPS based AuthZ service, it also provides the function of mapping user/requestor ID to the local pool account. In this respect current LCAS/LCMAPS implementation can be treated as a stateful PDP that combines the policy decision function and partly policy enforcement functions.

In contrary the native XACML PDP is a stateless PDP but can use Obligations policy enforcement mechanism to provide instructions to PEP on the further policy decision enforcement.

One of the key functional elements in the most of currently used Grid AuthZ models is the gLExec that acts as a gateway between Grid services and Unix executing environment and provides mapping between Grid user/requestor identity and Unix identity. In the context of the discussed here SAML-XACML protocol and common SAML-XACML libraries, the gLExec can potentially operate as an obligated account mapping enforcement mechanism.

2.2 Obligations enforcement scenarios

This section explains different scenarios of enforcing obligations:

- Obligations are enforced by PEP at the time of receiving obligated AuthZ decision from PDP;
- Obligations are enforced at later time when the requestor accesses the resource or service
- Obligations are enforced before or after the resource or service delivered/accessed/consumed

First two scenarios will be discussed in details in the next version of the document. Now we can just mention that the first scenario is agreed for the initial implementation of the SAML-XACML library. Such functionality can be supported by ObligationHandlers that can be called either from the PEP or from the SAML-XACML interface modules that handles request/response messages.

Although allowing simple solution/implementation, the first method will have problems when enforcing Obligations for later access/use of the reserved service. It was discussed that to allow Obligations enforcement at later time the AuthZ ticket or assertion can be used that contain all necessary information about the AuthZ request/response context. In this case AuthZ ticket must be properly secured with the XML signature and additionally encrypted. AuthZ ticket can use the SAML assertion containing XACMLAuthzDecisionStatement.

Additionally, two other scenarios can be considered to reflect currently used/implemented interoperation between stateful and stateless AuthZ service components like in case of LCAS/LCMAP and G-PBox interaction to combine flexible XACML based site/resource policy evaluation by G-PBox and required for most of Grid resources local pool account mapping by LCAS/LCMAPS.

2.3 Policy Obligations and their use in the typical Grid oriented authorisation scenarios

Obligation is one of the authorisation policy enforcement mechanisms that allows adding AuthZ decision enforcement components that can not be defined in the policy at the moment of making policy decision by the PDP, or may not be known to the PDP or policy administrator/writer

Suggested functionality that can be achieved with using obligations as a policy definition and enforcement mechanism:

- Account mapping
- Quota assignment
- Service combination with implied conditions (e.g., computing and storage resources)
- Usable resources/quota

[Additional information and design suggestions will be provide]

3 Obligations processing model and suggested implementation

3.1 Generic Authorisation Request processing flow and Obligations Handling Reference Model (OHRM)

Figure 3.1 below illustrates generic model for processing obligations in Site-central AuthZ service (SCAS). Described here processing model is compliant to the model used in XACML (refer to XACML2.0 standard or see Appendix) but adds Web services and AuthZ callout protocol details and specifically focuses on Obligations handling flow.

Note. This section discusses uses generic X.812 access control framework [X.812] and compliant to it generic AAA Authorisation framework [GAAA-AuthZ] AuthZ model to introduce Obligations Handling Reference Model (OHRM). Next section provides OHRM application to the gJAF AuthZ processing flow.

A number of assumptions are made to reflect possible options in AuthZ service infrastructure implementation and different type of Obligations both stateful and stateless that are concerned with assigning pool accounts, enforcing quotas, controlling usable resource (e.g., number of resource access, purchased video/music listening time, etc.), logging and accounting.

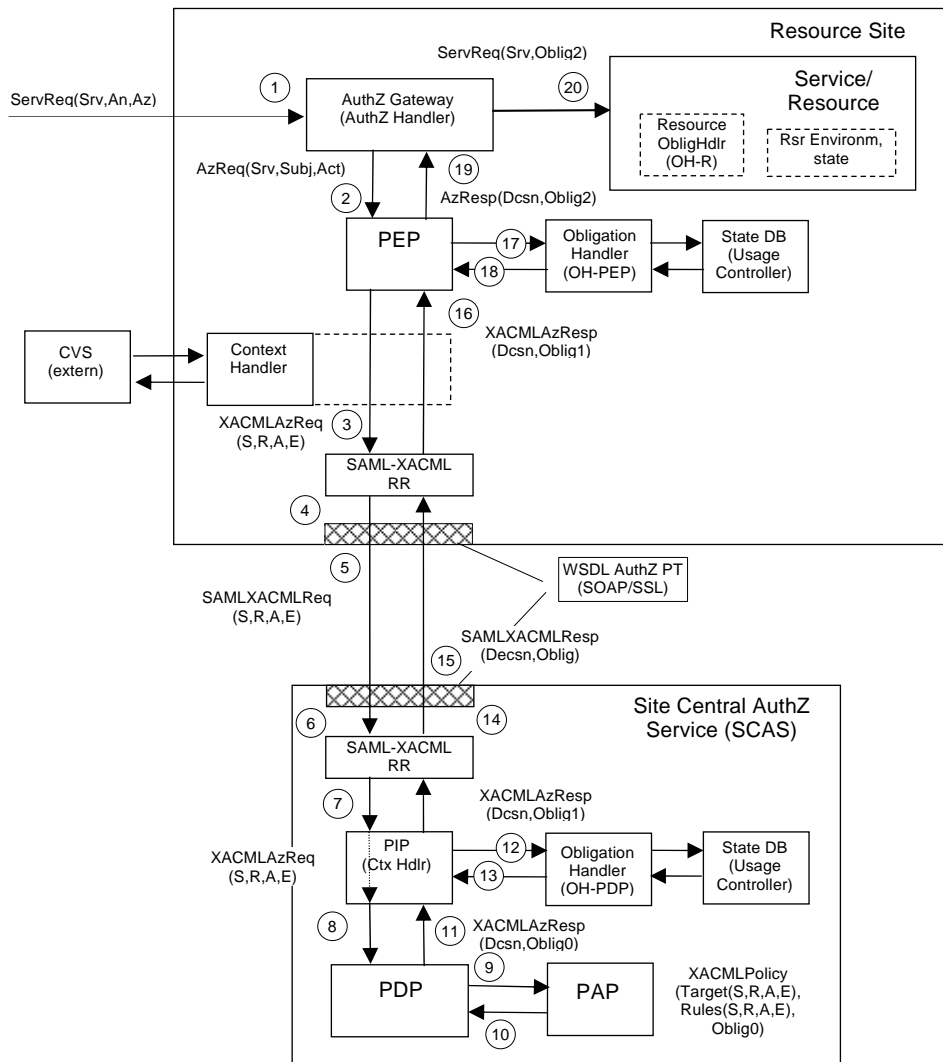


Figure 3.1. Generic Authorisation dataflow and Obligations handling in distributed AuthZ service.

It is important to notice that Obligations are the integral part of the policy and typically included into the policy at the stage of its creation by the policy administrator or resource owner. For the manageability purpose, policy is considered stateless and statefulness of Obligations is achieved by the Obligation handlers. The obligations enforcement process can be resulted either in modifying ServiceRequest (e.g., map from Subject to account name/type) or changing Resource/system state or environment.

For the general (stateful) Obligations handling process there can be distinguished the following stages (note: not all stages are necessary to be implemented in one use case by they may exist in different cases):

Obligation0 = tObligation => Obligation1 (“OK?”, (Attributes1 v Environments1)) => Obligation2 (“OK?”, (Attributes2 v Environments2)) => Obligation3 (Attributes3 v Environments3)

1) Obligation0 – (stateless) Obligations are returned by the PDP in a form as they are written in the policy. These obligations can be also considered as a kind of templates or instructions, tObligation. (Important to mention that due to security reason Obligations format and semantics should not use executable code or reference to locally executed commands).

2) Obligation1 or Obligation 2 – Obligations have been handled by Obligation handler at the SCAS/PDP side or at the PEP side, depending on implementation. In this case templates or instructions of the Obligation0 are replaced with the real attributes in Obligation1, e.g. in a form of “name-value” pair. During this stage, the Obligation handler can actually enforce Obligations or modify Obligations and send them further for enforcement by the Resource. The result of Obligations processing/enforcement, can be returned in a form of modified AuthzResponse (Obligation1) or in a form of global Resource environment changes that will be taken into account at the time when the requested service/resource are provided or delivered. In both cases (and specifically in the last case) Obligation handler should return notification about fulfilled

obligated actions, e.g. in a form of Boolean value “False” or “True”, which will be taken into account by PEP or other processing module to finally permit or deny service request by PEP.

Note. Option with Obligation1 handling at the SCAS or PDP side is introduced to illustration a case when we need to implement a stateful PDP/SCAS. This should not be considered as XACML specification violation as distinguishing between PEP and PDP functions in the generic Obligations handling model is based on what module actually makes policy based request evaluation.

3) Obligation3 – This is the final stage when Obligations actually take effect, which can be defined as Obligations “termination”. This is done by the Resource itself or by services managed/controlled by the Resource.

Note. In this general discussion we are not considering possible logical or time wise sequence of enforcing Obligations, but this is a topic of recent discussion at “xacml-dev” and “ogsa-authz” mailing lists.

3.2 Authorisation Request processing flow and Obligations handling model in gJAF

Figure 3.2 below illustrates possible Authorisation dataflow and Obligations handling when using external AuthZ callout from gJAF (and inheritably from the GT-AuthZ framework) to Site-central AuthZ service (SCAS).

All discussions from the previous section are applicable to this model. The figure also illustrates an option of flexible Obligation handlers registration at the ExtPDP Callout module which in the proposed module can be treated as a part of virtual PEP module. It is also considered that Obligations enforcement can be resulted either in changing SecurityContext or global Resource environments.

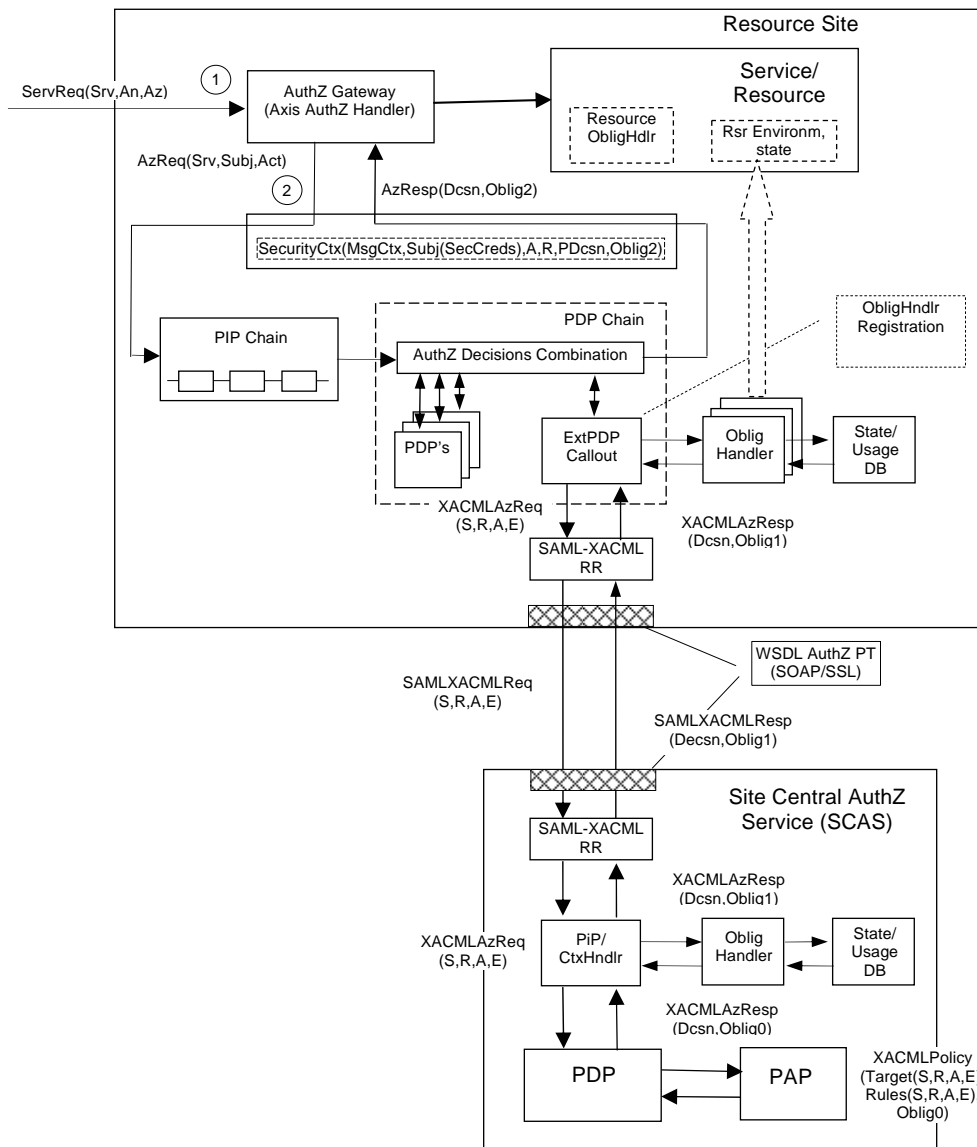


Figure 3.2. Authorisation dataflow and Obligations handling when using external AuthZ callout from gJAF or GT-AuthZ framework.

4 Obligations expression in the XACML policy and proposed conventions

4.1 Obligations definition and expression in the XACML policy

From XACML 2.0 specification section “2.12. Actions performed in conjunction with enforcement” (lines 557-566):

In many applications, policies specify actions that **MUST** be performed, either instead of, or in addition to, actions that **MAY** be performed. This idea was described by Sloman [Sloman94]. XACML provides facilities to specify actions that **MUST** be performed in conjunction with policy evaluation in the `<Obligations>` element. This idea was described as a provisional action by Kudo [Kudo00]. There are no standard definitions for these actions in version 2.0 of XACML. Therefore, bilateral agreement between a PEP and the PEP that will enforce its policies is required for correct interpretation. PEPs that conform with v2.0 of XACML are required to deny *access* unless they understand and can discharge all of the `<Obligations>` elements associated with the *applicable policy*. `<Obligations>` elements are returned to the PEP for enforcement.

For more reference information from the XACML specification refer to Appendix B.

4.2 Examples of expressing and handling Obligations

The examples in this section illustrate how Obligations can be expressed in the XACML policy format models an idea to communicate PEP Obligations handling capability to the PDP in the Environment element. However, to make it possible to select the applicable policy based on returned Obligations, we need to put explicit values of the ObligationId's into the policy Environment matching expression.

4.2.1 Example maps UID and GUD in pool accounts and communicates PEP capabilities as a Resource attribute "peptype"

a) Policy example that maps-to/enforces UID and GUD in pool accounts and use simple way to define PEP's obligation processing functionality as additional Resource/Attribute by specifying "peptype"

```
<?xml version="1.0" encoding="UTF-8"?>
<Policy
  xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
  xmlns:xacml-context="urn:oasis:names:tc:xacml:2.0:context:schema:os"
  xmlns:md="http://www.medico.com/schemas/record"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:policy:schema:os
    access_control-xacml-2.0-policy-schema-os.xsd"
  PolicyId="urn:oasis:names:tc:xacml:2.0:scas-policy:example007:policy"
  RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:deny-overrides">
  <Description>
  Example007 - Test for policy selection by PEP type.
  </Description>
  <PolicyDefaults>
    <XPathVersion>http://www.w3.org/TR/1999/Rec-xpath-19991116</XPathVersion>
  </PolicyDefaults>
  <Target>
    <Subjects>
      <Subject>
        <SubjectMatch
          MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
          <AttributeValue
            DataType="http://www.w3.org/2001/XMLSchema#string">
            VO-EGEE</AttributeValue>
          <SubjectAttributeDesignator
            SubjectCategory="urn:oasis:names:tc:xacml:1.0:subject:-category:access-subject"
            AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-vo"
            DataType="http://www.w3.org/2001/XMLSchema#string"/>
          </SubjectMatch>
        </Subject>
      </Subjects>
    <Actions>
      <AnyAction/>
    </Actions>
    <Resources>
      <Resource>
        <ResourceMatch
          MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
          <AttributeValue
            DataType="http://www.w3.org/2001/XMLSchema#string">http://nikhef.nl/VO-EGEE/CE01
          </AttributeValue>
          <ResourceAttributeDesignator
            AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
            DataType="http://www.w3.org/2001/XMLSchema#URI"/>
          </ResourceMatch>
        <ResourceMatch
          MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
          <AttributeValue
            DataType="http://www.w3.org/2001/XMLSchema#string">GT4-CE
          </AttributeValue>
          <ResourceAttributeDesignator
            AttributeId="urn:some:path:peptype"
            DataType="http://www.w3.org/2001/XMLSchema#String"/>
          </ResourceMatch>
        </Resource>
      </Resources>
    </Target>
  </Rule>
  <Rule
    RuleId="urn:oasis:names:tc:xacml:2.0:scas-policy:example007:rule"
    Effect="Permit">
    <Description>
    User with role "researcher" from VO "EGEE" can access Resource "CE".
    </Description>
  </Rule>
</Policy>
```

```

<Subjects>
  <AnySubject/>
</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">SubmitJob</AttributeValue>
        <ActionAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-
id"
          DataType="http://www.w3.org/2001/XMLSchema#string"/>
        </ActionMatch>
      </Action>
    </Actions>
  </Target>
  <Condition FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-at-least-one-member-of">
    <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-bag">
      <AttributeValue
        DataType="http://www.w3.org/2001/XMLSchema#string">researcher</AttributeValue>
      </Apply>
      <SubjectAttributeDesignator
        AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-role"
        DataType="http://www.w3.org/2001/XMLSchema#string"
        Issuer="EGEEAttributeIssuer"/>
    </Condition>
  </Rule>
</Obligations>
  <Obligation
    ObligationId="urn:oasis:names:tc:xacml:2.0:scas-policy:example007:policy:obligation.UID"
    FulfillOn="Permit">
      <AttributeAssignment
        AttributeId="urn:oasis:names:tc:xacml:1.0:example:attribute:access-subject"
        DataType="http://www.w3.org/2001/XMLSchema#string">
          &lt;SubjectAttributeDesignator
            AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
            DataType="http://www.w3.org/2001/XMLSchema#string"/&gt;
          </AttributeAssignment>
        <!-- This is actual account attribute/name to which it should be mapped -->
        <AttributeAssignment
          AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:poolaccount"
          DataType="http://www.w3.org/2001/XMLSchema#string">
            &lt;PoolAccountDesignator
              AttributeId="http://glite.egee.org/JRA1/Authz/XACML/obligation/poolaccount"
              DataType="http://www.w3.org/2001/XMLSchema#string"/&gt;
            <AttributeValue
              DataType="http://www.w3.org/2001/XMLSchema#string">egee-pool-next-available
            </AttributeValue>
          </AttributeAssignment>
        </Obligation>
      </Obligation>
    </Obligation>
  </Obligations>
</Policy>

```

b) Corresponding Request that contains additional Resource/Attribute information “peptype” that indicates specific PEP type/functionality.

```

<?xml version="1.0" encoding="UTF-8"?>
<Request
  xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:context:schema:os
  access_control-xacml-2.0-context-schema-os.xsd">
  <Subject>
    <Attribute

```

```

        AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
        DataType="http://www.w3.org/2001/XMLSchema#string">
        <AttributeValue>Wim Huizinga</AttributeValue>
    </Attribute>
    <Attribute
        AttributeId="urn:oasis:names:tc:xacml:2.0:subject:subject-vo"
        DataType="http://www.w3.org/2001/XMLSchema#string">
        <AttributeValue>VO-EGEE</AttributeValue>
    </Attribute>
    <Attribute
        AttributeId="urn:oasis:names:tc:xacml:2.0:subject:subject-role"
        DataType="http://www.w3.org/2001/XMLSchema#string">
        <AttributeValue>researcher</AttributeValue>
    </Attribute>
</Subject>
<Resource>
    <Attribute
        AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
        DataType="http://www.w3.org/2001/XMLSchema#anyURI">
        <AttributeValue>http://nikhef.nl/VO-EGEE/CE01</AttributeValue>
    </Attribute>
    <Attribute
        AttributeId="urn:some:path:peptype"
        DataType="http://www.w3.org/2001/XMLSchema#string">
        <AttributeValue>CE-GT4</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>SubmitJob</AttributeValue>
    </Attribute>
</Action>
<Environment/>
</Request>

```

4.2.2 Example maps UID and GUD in pool accounts and communicates PEP capabilities by referring to the Request context information

a) Example policy the maps-to/enforces UID and GUD in pool accounts and uses explicit reference to the type of Obligations supported by PEP by referring to the Request context information.

```

<?xml version="1.0" encoding="UTF-8"?>
<Policy
    xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
    xmlns:xacml-context="urn:oasis:names:tc:xacml:2.0:context:schema:os"
    xmlns:md="http://www.medico.com/schemas/record"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:policy:schema:os
        access_control-xacml-2.0-policy-schema-os.xsd"
    PolicyId="urn:oasis:names:tc:xacml:2.0:scas-policy:example007:policy"
    RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:deny-overrides">
    <Description>
    Example007 - Test for policy selection by supported obligations in Environment element.
    </Description>
    <PolicyDefaults>
        <XPathVersion>http://www.w3.org/TR/1999/Rec-xpath-19991116</XPathVersion>
    </PolicyDefaults>
    <Target>
        <Subjects>
            <Subject>
                <SubjectMatch
                    MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
                        DataType="http://www.w3.org/2001/XMLSchema#string">
                        VO-EGEE</AttributeValue>
                    <SubjectAttributeDesignator
                        SubjectCategory="urn:oasis:names:tc:xacml:1.0:subject:-category:access-subject"
                        AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-vo"
                        DataType="http://www.w3.org/2001/XMLSchema#string"/>
                    </SubjectMatch>
                </Subject>
            </Subjects>
        <Actions>
            <AnyAction/>
        </Actions>
    </Target>

```

```

<Resources>
  <Resource>
    <ResourceMatch
      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
    <AttributeValue
      DataType="http://www.w3.org/2001/XMLSchema#string">http://nikhef.nl/VO-EGEE/CE01
    </AttributeValue>
    <ResourceAttributeDesignator
      AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
      DataType="http://www.w3.org/2001/XMLSchema#URI"/>
M    </ResourceMatch>
  </Resource>
</Resources>
<Environments>
  <Environment>
    <EnvironmentMatch
      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
    <AttributeValue
      DataType="http://www.w3.org/2001/XMLSchema#string">
      obligation.UID
    </AttributeValue>
    <AttributeSelector
      RequestContextPath="./xacml-context:Environment/xacml-context:Attribute/xacml-
context:AttributeValue/scas:PEPconfig/scas:SupportedObligations/scas:ObligationId/@ObligationId"
      MustBePresent="true"
      DataType="http://www.w3.org/2001/XMLSchema#string"/>
    </EnvironmentMatch>
  </Environment>
  <Environment>
    <EnvironmentMatch
      MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
    <AttributeValue
      DataType="http://www.w3.org/2001/XMLSchema#string">
      obligation.GID
    </AttributeValue>
    <AttributeSelector
      RequestContextPath="./xacml-context:Environment/xacml-context:Attribute/xacml-
context:AttributeValue/scas:PEPconfig/scas:SupportedObligations/scas:ObligationId/@ObligationId"
      MustBePresent="true"
      DataType="http://www.w3.org/2001/XMLSchema#string"/>
    </EnvironmentMatch>
  </Environment>
</Environments>
</Target>
<Rule
  RuleId="urn:oasis:names:tc:xacml:2.0:scas-policy:example007:rule"
  Effect="Permit">
  <Description>
    User with role "researcher" from VO "EGEE" can access Resource "CE".
  </Description>
  <Target>
    <Subjects>
      <AnySubject/>
    </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">SubmitJob</AttributeValue>
        <ActionAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-
id"
          DataType="http://www.w3.org/2001/XMLSchema#string"/>
        </ActionMatch>
      </Action>
    </Actions>
  </Target>
  <Condition FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-at-least-one-member-of">
    <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-bag">
      <AttributeValue
        DataType="http://www.w3.org/2001/XMLSchema#string">researcher</AttributeValue>
      </Apply>
      <SubjectAttributeDesignator
        AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-role"
        DataType="http://www.w3.org/2001/XMLSchema#string"
        Issuer="EGEEAttributeIssuer"/>
    </Condition>
  </Rule>

```

```

<Obligations>
  <Obligation
    ObligationId="urn:oasis:names:tc:xacml:2.0:scas-policy:example007:policy:obligation.UID"
    FulfillOn="Permit">
    <AttributeAssignment
      AttributeId="urn:oasis:names:tc:xacml:1.0:example:attribute:access-subject"
      DataType="http://www.w3.org/2001/XMLSchema#string">
      &lt;SubjectAttributeDesignator
        AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
        DataType="http://www.w3.org/2001/XMLSchema#string"/&gt;
      </SubjectAttributeDesignator>
    </AttributeAssignment>
    <!-- This is actual account attribute/name to which it should be mapped -->
    <AttributeAssignment
      AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:poolaccount"
      DataType="http://www.w3.org/2001/XMLSchema#string">
      &lt;PoolAccountDesignator
        AttributeId="http://glite.egee.org/JRA1/Authz/XACML/obligation/poolaccount"
        DataType="http://www.w3.org/2001/XMLSchema#string"/&gt;
      </PoolAccountDesignator>
    <AttributeValue
      DataType="http://www.w3.org/2001/XMLSchema#string">egee-pool-next-available
    </AttributeValue>
    </AttributeAssignment>
  </Obligation>
  <Obligation
    ObligationId="urn:oasis:names:tc:xacml:2.0:scas-policy:example007:policy:obligation.GID"
    FulfillOn="Permit">
    <AttributeAssignment
      AttributeId="urn:oasis:names:tc:xacml:1.0:scas-policy:subject:subject-group"
      DataType="http://www.w3.org/2001/XMLSchema#string">
      GID-researchers
    </AttributeAssignment>
  </Obligation>
</Obligations>
</Policy>

```

Corresponding Request message that contains information about PEP Obligations capability in the Environment element

```

<?xml version="1.0" encoding="UTF-8"?>
<Request
  xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:context:schema:os
  access_control-xacml-2.0-context-schema-os.xsd">
  <Subject>
    <Attribute
      AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
      DataType="http://www.w3.org/2001/XMLSchema#string">
      <AttributeValue>Wim Huizinga</AttributeValue>
    </Attribute>
    <Attribute
      AttributeId="urn:oasis:names:tc:xacml:2.0:subject:subject-vo"
      DataType="http://www.w3.org/2001/XMLSchema#string">
      <AttributeValue>VO-EGEE</AttributeValue>
    </Attribute>
    <Attribute
      AttributeId="urn:oasis:names:tc:xacml:2.0:subject:subject-role"
      DataType="http://www.w3.org/2001/XMLSchema#string">
      <AttributeValue>researcher</AttributeValue>
    </Attribute>
  </Subject>
  <Resource>
    <Attribute
      AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
      DataType="http://www.w3.org/2001/XMLSchema#anyURI">
      <AttributeValue>http://nikhef.nl/VO-EGEE/CE01</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>SubmitJob</AttributeValue>
    </Attribute>
  </Action>
  <Environment>
    <Attribute
      AttributeId="urn:oasis:names:tc:xacml:2.0:scas-policy:pep-config:obligations"
      DataType="http://www.w3.org/2001/XMLSchema#xml">
    <AttributeValue>
      <scas:PEPconfig>

```

```

    <scas:SuportedObligations>
      <scas:Obligation ObligationId="obligation.UID" />
      <scas:Obligation ObligationId="obligation.GID" />
    </scas:SuportedObligations>
    <scas:PEPconfig>
  </AttributeValue>
</Attribute>
</Environment>
</Request>

```

c) Example Response message with returned Obligations containing obligations to assign/map to UID and GID

```

<?xml version="1.0" encoding="UTF-8"?>
<Response xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:context:schema:os access_control-xacml-2.0-
context-schema-os.xsd">
  <Result ResourceId=" http://nikhef.nl/VO-EGEE/CE01">
    <Status>
      <StatusCode Value="OK" />
      <StatusDetail>DecisionID</StatusDetail>
      <StatusMessage>Request is successful</StatusMessage>
    </Status>
    <Decision>Permit</Decision>
    <xacml:Obligations>
      <xacml:Obligation
        ObligationId="urn:oasis:names:tc:xacml:2.0:scas-policy:example007:policy:obligation.UID"
        FulfillOn="Permit">
        <xacml:AttributeAssignment
          AttributeId="urn:oasis:names:tc:xacml:1.0:example:attribute:access-subject"
          DataType="http://www.w3.org/2001/XMLSchema#string">
            &lt;SubjectAttributeDesignator
              AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
              DataType="http://www.w3.org/2001/XMLSchema#string" /&gt;
            </xacml:AttributeAssignment>
<!-- This is actual account attribute/name to which it should be mapped -->
          <xacml:AttributeAssignment
            AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:poolaccount"
            DataType="http://www.w3.org/2001/XMLSchema#string">
              &lt;PoolAccountDesignator
                AttributeId="http://glite.egee.org/JRA1/Authz/XACML/obligation/poolaccount"
                DataType="http://www.w3.org/2001/XMLSchema#string" /&gt;
              <xacml:AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
                egee-pool-next-available
              </xacml:AttributeValue>
            </xacml:AttributeAssignment>
          </xacml:Obligation>
        </xacml:Obligation
          ObligationId="urn:oasis:names:tc:xacml:2.0:scas-policy:example007:policy:obligation.GID"
          FulfillOn="Permit">
          <xacml:AttributeAssignment
            AttributeId="urn:oasis:names:tc:xacml:1.0:scas-policy:subject:subject-group"
            DataType="http://www.w3.org/2001/XMLSchema#string">
              GID-researchers
            </xacml:AttributeAssignment>
          </xacml:Obligation>
        </xacml:Obligations>
      </Result>
    </Response>

```

d) Response message after transforming by stateful pool accounts manager or UID Obligation handler – UID attribute value changed from declarative “egee-pool-next-available” to a specific pool account “egee-pool01”.

```

<?xml version="1.0" encoding="UTF-8"?>
<Response xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os"
xmlns:xacml="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:context:schema:os access_control-xacml-2.0-
context-schema-os.xsd">
  <Result ResourceId=" http://nikhef.nl/VO-EGEE/CE01">
    <Status>
      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok" />
    </Status>
    <Decision>Permit</Decision>
    <xacml:Obligations>

```

```

<xacml:Obligation
  ObligationId="urn:oasis:names:tc:xacml:2.0:scas-policy:example007:policy:obligation.UID"
  FulfillOn="Permit">
<!-- This the type of attribute to which Obligation is applied, i.e. account mapping -->
  <xacml:AttributeAssignment
    AttributeId="urn:oasis:names:tc:xacml:1.0:example:attribute:access-subject"
    DataType="http://www.w3.org/2001/XMLSchema#string">
      &lt;SubjectAttributeDesignator
        AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
        DataType="http://www.w3.org/2001/XMLSchema#string"/&gt;
      </xacml:AttributeAssignment>
<!-- This is actual account attribute/name to which it should be mapped -->
  <xacml:AttributeAssignment
    AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:poolaccount"
    DataType="http://www.w3.org/2001/XMLSchema#string">
      &lt;PoolAccountDesignator
        AttributeId="http://glite.egee.org/JRA1/Authz/XACML/obligation/poolaccount"
        DataType="http://www.w3.org/2001/XMLSchema#string"/&gt;
      <xacml:AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
        egee-pool01
      </xacml:AttributeValue>
    </xacml:AttributeAssignment>
  </xacml:Obligation>
xacml:Obligation
  ObligationId="urn:oasis:names:tc:xacml:2.0:scas-policy:example007:policy:obligation.GID"
  FulfillOn="Permit">
  <xacml:AttributeAssignment
    AttributeId="urn:oasis:names:tc:xacml:1.0:scas-policy:subject:subject-group"
    DataType="http://www.w3.org/2001/XMLSchema#string">
    GID-researchers
  </xacml:AttributeAssignment>
</xacml:Obligation>
</xacml:Obligations>
</Response>

```

4.3 Proposed Obligation expression conventions

This section will provide suggestions and considerations about Obligations expression and ObligationId format in particular.

ObligationId format should be based on agreed namespace identifier for the target project or use cases and can use either URN or URI form. For the purpose of this document the following namespace identifiers are suggested (however, final choice will depend on the project consensus and required formal decisions):

glite:security:authz:(policy | policy:obligation)

glite:security:authn

http://glite.org/security/authorisation/

For the management and also development/deployment purposes it should be allowed to create the following namespace sub-trees/branches:

orgname/projname

servicename

example

test

The following are examples of the attributes and obligations identifiers:

1) using SAML/XACML URN style

urn:oasis:names:tc:xacml:2.0:glite:security:authz:policy:obligation:obligation.UID

urn:oasis:names:tc:xacml:2.0:glite:security:authz:example007:policy:obligation:obligation.UID

urn:oasis:names:tc:xacml:2.0:glite:security:authz:EGEE:policy:obligation:obligation.UID

2) using general URI style

<http://glite.org/security/authorisation/policy/obligation/obligation.UID>

<http://glite.org/security/authorisation/CNAF/policy/obligation/obligation.UID>

<http://glite.org/security/authorisation/CREAM/policy/obligation/obligation.UID>

5 Obligation Handling API

To be added, possibly after initial implementations.

6 Conformance test

This section will discuss requirements to and describe the conformance test to ensure interoperability of different SAML-XACML protocol implementations and Obligations expression and handling models.

As an initial input, the registered Obligation types will be used (see document at <http://home.fnal.gov/~garzogli/privilege/AuthZInterop/tmp/AuthZInterop%20XACML%20Profile%20v0.3.doc>)

- UID + GID
- Optional multiple 2ndary GIDs
- Optional AFS token (type string)
- Username (for CE)
- UID + GID (common w/ EGEE)
- RootPath + HomeDir (gPlazma)
- Priorities (gPlazma)
- File creation mask + directory creation mask

References

- [1] *Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0*, OASIS Standard, 15 March 2005. [Online]. Available: <http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf>
- [2] *SAML 2.0 Profile of XACML 2.0, Version 2. Working Draft 2*, 26 June 2006. [Online]. Available: <http://docs.oasis-open.org/xacml/2.0/xacml-2.0-profile-saml2.0-v2.zip>
- [3] *eXtensible Access Control Markup Language (XACML) Version 2.0*, OASIS Standard, 1 February 2005. [Online]. Available: http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf
- [4] Chadwick, D., "Use of WS-TRUST and SAML to access a CVS". OGSA-AUTHZ WG Draft. [Online]. Available: https://forge.gridforum.org/sf/docman/do/downloadDocument/projects.ogsa-authz/docman.root.authz_service/doc9011/1
- [5] "Multiple resource profile of XACML 2.0", OASIS Standard, 1 February 2005, available from http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-mult-profile-spec-os.pdf
- [6] "Hierarchical resource profile of XACML 2.0", OASIS Standard, 1 February 2005, available from http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-hier-profile-spec-os.pdf
- [7] *Core and hierarchical role based access control (RBAC) profile of XACML v2.0*, OASIS Standard, 1 February 2005. [Online]. Available: http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-rbac-profile1-spec-os.pdf
- [8] "XACML 3.0 administrative policy," OASIS Draft, 10 December 2005. [Online]. Available from http://docs.oasis-open.org/access_control
- [9] ITU-T Rec. X.812 (1995) | ISO/IEC 10181-3:1996, Information technology - Open systems interconnection - Security frameworks in open systems: Access control framework. [Online]. Available: http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-X.812-199511-I!!PDF-E&type=items
- [10] RFC 2904 - "AAA Authorization Framework" J. Vollbrecht, P. Calhoun, S. Farrell, L. Gommans, G. Gross, B. de Bruijn, C. de Laat, M. Holdrege, D. Spence, August 2000 - <ftp://ftp.isi.edu/in-notes/rfc2904.txt>
- [11] GFD.38 Conceptual Grid Authorization Framework and Classification. M. Lorch, B. Cowles, R. Baker, L. Gommans, P. Madsen, A. McNab, L. Ramakrishnan, K. Sankar, D. Skow, M. Thompson - <http://www.ggf.org/documents/GWD-I-E/GFD-I.038.pdf>
- [12] Demchenko Y., et al., Dynamic security context management in Grid-based applications, *Future Generation Computer Systems* (2007), doi:10.1016/j.future.2007.07.015
- [13] RFC2748: The COPS (Common Open Policy Service) Protocol, Edited Durham, D., January 2000. - <http://www.ietf.org/rfc/rfc2748.txt>
- [14] RFC2753: A Framework for Policy-based Admission Control, January 2000. - <http://www.ietf.org/rfc/rfc2753.txt>

- [15] Xinwen Zhang, Masayuki Nakae, Michael J. Covington, and Ravi Sandhu, A Usage-based Authorization Framework for Collaborative Computing Systems, in the proceedings of ACM Symposium on Access Control Models and Technologies (SACMAT), 2006.
- [16] Deliverable DJ5.2.2: GÉANT2 Authorisation and Authentication Infrastructure (AAI) Architecture. - <http://www.geant2.net/upload/pdf/GN2-05-192v6.pdf>
- [17] Deliverable DJ5.2.3,2: Best Practice Guide - AAI Cookbook - Second Edition Guidelines for Connecting to the eduGAIN AA Infrastructure. - http://www.geant2.net/upload/pdf/GN2-07-023v4-DJ5-2-3_2_Best_Practice_Guide-AAI_Cookbook-Second_Edition.pdf

Appendix A - Obligations definition and expression in the XACML specification

[XACML 5.45.] Element <Obligation>

The <Obligation> element SHALL contain an identifier for the *obligation* and a set of *attributes* that form arguments of the action defined by the *obligation*. The FulfillOn attribute SHALL indicate the *effect* for which this *obligation* must be fulfilled by the *PEP*.

```
<xs:element name="Obligation" type="xacml:ObligationType"/>
<xs:complexType name="ObligationType">
<xs:sequence>
<xs:element ref="xacml:AttributeAssignment" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
<xs:attribute name="ObligationId" type="xs:anyURI" use="required"/>
<xs:attribute name="FulfillOn" type="xacml:EffectType" use="required"/>
</xs:complexType>
```

The <Obligation> element is of **ObligationType** complexType. See Section 7.14 for a description of how the set of *obligations* to be returned by the *PDP* is determined. The <Obligation> element contains the following elements and attributes:

ObligationId [Required]

Obligation identifier. The value of the *obligation* identifier SHALL be interpreted by the *PEP*.

FulfillOn [Required]

The **effect** for which this *obligation* must be fulfilled by the *PEP*.

<AttributeAssignment> [Optional]

Obligation arguments assignment. The values of the *obligation* arguments SHALL be interpreted by the *PEP*.

[XACML-5.46.] Element <AttributeAssignment>

The <AttributeAssignment> element is used for including arguments in *obligations*. It SHALL contain an AttributeId and the corresponding *attribute* value, by extending the AttributeValueType type definition. The <AttributeAssignment> element MAY be used in any way that is consistent with the schema syntax, which is a sequence of <xs:any> elements. The value specified SHALL be understood by the *PEP*, but it is not further specified by XACML. See Section 7.14. Section 4.2.4.3 provides a number of examples of arguments included in *obligations*.

```
<xs:element name="AttributeAssignment" type="xacml:AttributeAssignmentType"/>
<xs:complexType name="AttributeAssignmentType" mixed="true">
<xs:complexContent>
<xs:extension base="xacml:AttributeValueType">
<xs:attribute name="AttributeId" type="xs:anyURI" use="required"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
```

The <AttributeAssignment> element is of **AttributeAssignmentType** complex type. The

<AttributeAssignment> element contains the following attributes:

AttributeId [Required]

The **attribute** Identifier.

[XACML-7.14. Obligations]

A *policy* or *policy set* may contain one or more *obligations*. When such a *policy* or *policy set* is evaluated, an *obligation* SHALL be passed up to the next level of evaluation (the enclosing or referencing *policy*, *policy set* or *authorization decision*) only if the *effect* of the *policy* or *policy set* being evaluated matches the value of the FulfillOn attribute of the *obligation*.

As a consequence of this procedure, no *obligations* SHALL be returned to the *PEP* if the *policies* or *policy sets* from which they are drawn are not evaluated, or if their evaluated result is "Indeterminate" or "NotApplicable", or if the *decision* resulting from evaluating the *policy* or *policy set* does not match the *decision* resulting from evaluating an enclosing *policy set*.

If the *PDP*'s evaluation is viewed as a tree of *policy sets* and *policies*, each of which returns "Permit" or "Deny", then the set of *obligations* returned by the *PDP* to the *PEP* will include only the *obligations* associated with those paths where the *effect* at each level of evaluation is the same as the *effect* being returned by the *PDP*. In situations where any lack of determinism is unacceptable, a deterministic combining algorithm, such as ordered-deny-overrides, should be used.

<<To be extended>>

Appendix B - XACML Obligations expression examples

The following contains examples of the obligations expression provided for reference purpose only.

a) Example <Obligations> from XACML2.0 specification:

```
<Obligations>
<Obligation
  ObligationId="urn:oasis:names:tc:xacml:example:obligation:email"
  FulfillOn="Permit">
  <AttributeAssignment
    AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:mailto"
    DataType="http://www.w3.org/2001/XMLSchema#string">
    &lt;AttributeSelector
      RequestContextPath="//md:/record/md:patient/md:patientContact/md:email"
      DataType="http://www.w3.org/2001/XMLSchema#string"/&gt;
    </AttributeSelector>
  </AttributeAssignment>
  <AttributeAssignment
    AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:text"
    DataType="http://www.w3.org/2001/XMLSchema#string">
    Your medical record has been accessed by:
  </AttributeAssignment>
  <AttributeAssignment
    AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:text"
    DataType="http://www.w3.org/2001/XMLSchema#string">
    &lt;SubjectAttributeDesignator
      AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
      DataType="http://www.w3.org/2001/XMLSchema#string"/&gt;
    </SubjectAttributeDesignator>
  </AttributeAssignment>
</Obligation>
</Obligations>
</Policy>
```

b) Example <Obligations> from OGSA-AuthZ specification "Use of XACML Request Context to access a PDP" (edited by David Chadwick)

```
<?xml version="1.0" encoding="UTF-8"?>
<Response xmlns="urn:oasis:names:tc:xacml:2.0:context:schema:os"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:context:schema:os
  http://docs.oasis-open.org/xacml/xacml-core-2.0-context-schema-os.xsd">
  <Result ResourceId="12345">
    <Decision>Permit</Decision>
    <Status>
      <StatusCode Value="urn:oasis:names:tc:xacml:1.0:status:ok"/>
    </Status>
    <Obligations>
      <Obligation
        ObligationId="http://sec.cs.kent.ac.uk/GGF/XACML/obligation/example"
        FulfillOn="Permit">
          <AttributeAssignment
            AttributeId="http://sec.cs.kent.ac.uk/GGF/XACML/environment/balance"
            DataType="http://www.w3.org/2001/XMLSchema#integer">
            &lt;Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:integer-add"&gt;
              &lt;Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:integer-one-and-only"&gt;
                &lt;ActionAttributeDesignator AttributeId="http://sec.cs.kent.ac.uk/GGF/XACML/MRAM.get.size"
                  DataType="http://www.w3.org/2001/XMLSchema#integer"/&gt;
                &lt;/Apply>
              &lt;/Apply>
            &lt;/Apply>
            &lt;Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:integer-one-and-only"&gt;
              &lt;EnvironmentAttributeDesignator
                AttributeId="http://sec.cs.kent.ac.uk/GGF/XACML/environment/balance"
                DataType="http://www.w3.org/2001/XMLSchema#integer"/&gt;
              &lt;/EnvironmentAttributeDesignator>
            &lt;/Apply>
          &lt;/Apply>
        </AttributeAssignment>
      </Obligation>
    </Obligations>
  </Result>
</Response>
```

Note. Presented above way of expressing Obligations may have serious security concerns because of including executive commands.

c) Example with pool accounts mapping in Grid

Note. Below options are provide only illustration how Obligations can be expressed in the XACML2.0 compliant format. This is not a goal of this document and section to make suggestions about preferred format, however it can be suggested that although Option 3 can bring possible flexibility it is not recommended way of expressing and communicating Obligations because of security concerns.

Option 1.

```
<!-- Obligations format option 1 (UID, GID explicitly mentioned as separate XML elements inside
AttributeAssignment element) -->
<Obligations>
<Obligation
  ObligationId="http://glite.egee.org/JRA1/Authz/XACML/obligation/map.poolaccount"
  FulfillOn="Permit">

<!-- This is a common part that specify to what kind of attribute the next 'map.to' action is
applied to -->
<AttributeAssignment
  AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute: requesting-subject"
  DataType="http://www.w3.org/2001/XMLSchema#string">
  &lt;SubjectAttributeDesignator
    AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
    DataType="http://www.w3.org/2001/XMLSchema#string"/&gt;
</AttributeAssignment>

<!-- This is actual account attribute/name to which it should be mapped -->
<AttributeAssignment
  AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:mapto"
  DataType="http://www.w3.org/2001/XMLSchema#string">
  &lt;UnixId
    DataType="http://www.w3.org/2001/XMLSchema#string"&gt;
    okoeroo&gt;UnixId&gt;
  &lt; GroupPrimary
    DataType="http://www.w3.org/2001/XMLSchema#string"&gt;
    computergroup&gt;GroupPrimary&gt;
  &lt;GroupSecondary
    DataType="http://www.w3.org/2001/XMLSchema#string"&gt;
    datagroup&gt;GroupSecondary&gt;
</AttributeAssignment>
</Obligation>
</Obligations>
```

Option 2.

```
<!-- Obligations format option 2 (UID, GID explicitly mentioned as one string in the
AttributeAssignment elements) -->
<Obligations>
<Obligation
  ObligationId="http://glite.egee.org/JRA1/Authz/XACML/obligation/map.poolaccount"
  FulfillOn="Permit">

<!-- This is a common part that specify to what kind of attribute the next 'map.to' action is
applied to -->
<AttributeAssignment
  AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:requesting-subject"
  DataType="http://www.w3.org/2001/XMLSchema#string">
  &lt;SubjectAttributeDesignator
    AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
    DataType="http://www.w3.org/2001/XMLSchema#string"/&gt;
</AttributeAssignment>

<!-- This is actual account attribute/name to which it should be mapped -->
<AttributeAssignment
  AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:poolaccount"
  DataType="http://www.w3.org/2001/XMLSchema#string">
  &lt;PoolAccountDesignator
    AttributeId="http://glite.egee.org/JRA1/Authz/XACML/obligation/poolaccount"
    UnixId="okoeroo" GroupPrimary="computergroup" GroupSecondary="datagroup"
    DataType="http://www.w3.org/2001/XMLSchema#string"/&gt;
</AttributeAssignment>
</Obligation>
</Obligations>
```

Option 3.

```

<!-- Obligations format option 3 (UID, GID are referred to by their uri in external gridmap file) -
->
<Obligations>
<Obligation
  ObligationId="http://glite.egee.org/JRA1/Authz/XACML/obligation/map.poolaccount"
  FulfillOn="Permit">

<!-- This is a common part that specify to what kind of attribute the next 'map.to' action is
applied to -->
<AttributeAssignment
  AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute: requesting-subject"
  DataType="http://www.w3.org/2001/XMLSchema#string">
  <&lt;SubjectAttributeDesignator
    AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
    DataType="http://www.w3.org/2001/XMLSchema#string"/&gt;&gt;
</AttributeAssignment>

<!-- This is actual account attribute/name to which it should be mapped -->
<AttributeAssignment
  AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:text"
  DataType="http://www.w3.org/2001/XMLSchema#string">
  <&lt;AttributeSelector
    GridMapPath="//gmap:/uid/gmap:primay/gmap:secondary"
    DataType="http://www.w3.org/2001/XMLSchema#string"/&gt;&gt; ;
</AttributeAssignment>
</Obligation>
</Obligations>

```

Appendix C - XACML Identifiers

Note: This excerpt from the XACML2.0 specification is provided for developers assistance.

XACML specification defines a number of attribute identifiers for common use in the "urn:oasis:names:tc:xacml:" namespace. The use of these identifiers in cases where it is possible will ensure better interoperability.

All other attribute identifiers which in many cases can be application specific are expected to use their own namespaces.

[Appendix B. XACML identifiers (normative)]

[B.1. XACML namespaces]

urn:oasis:names:tc:xacml:2.0:policy:schema:os

urn:oasis:names:tc:xacml:2.0:context:schema:os

[B.2. Access subject categories]

urn:oasis:names:tc:xacml:1.0:subject-category:access-subject

urn:oasis:names:tc:xacml:1.0:subject-category:recipient-subject

urn:oasis:names:tc:xacml:1.0:subject-category:intermediary-subject

urn:oasis:names:tc:xacml:1.0:subject-category:codebase

urn:oasis:names:tc:xacml:1.0:subject-category:requesting-machine

[B.3. Data-types]

These identifiers indicate data-types that are defined in the XACML specification Section A.2.

urn:oasis:names:tc:xacml:1.0:data-type:x500Name.

urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name

urn:oasis:names:tc:xacml:2.0:data-type:ipAddress

urn:oasis:names:tc:xacml:2.0:data-type:dnsName

The following data-type identifiers are defined by XML Schema.

<http://www.w3.org/2001/XMLSchema#string>

<http://www.w3.org/2001/XMLSchema#boolean>

<http://www.w3.org/2001/XMLSchema#integer>

<http://www.w3.org/2001/XMLSchema#double>

<http://www.w3.org/2001/XMLSchema#time>
<http://www.w3.org/2001/XMLSchema#date>
<http://www.w3.org/2001/XMLSchema#dateTime>
<http://www.w3.org/2001/XMLSchema#anyURI>
<http://www.w3.org/2001/XMLSchema#hexBinary>
<http://www.w3.org/2001/XMLSchema#base64Binary>

The following data-type identifiers correspond to the `dayTimeDuration` and `yearMonthDuration` data-types defined in [XF Sections 8.2.2 and 8.2.1, respectively].

<http://www.w3.org/TR/2002/WD-xquery-operators-20020816#dayTimeDuration>
<http://www.w3.org/TR/2002/WD-xquery-operators-20020816#yearMonthDuration>

[B.4. Subject attributes]

These identifiers indicate attributes of a subject. When used, they SHALL appear within a `<Subject>` element of the request context. They SHALL be accessed by means of a `<SubjectAttributeDesignator>` element, or an `<AttributeSelector>` element that points into a `<Subject>` element of the request context.

At most one of each of these attributes is associated with each subject. Each attribute associated with authentication included within a single `<Subject>` element relates to the same authentication event. This identifier indicates the name of the subject. The default format is `"http://www.w3.org/2001/XMLSchema#string"`. To indicate other formats, use the `DataType` attributes listed in B.3

`urn:oasis:names:tc:xacml:1.0:subject:subject-id`

This identifier indicates the subject's ID e.g. in a form of string, RFC822, or LDAP X.511 format

`urn:oasis:names:tc:xacml:1.0:subject:category`

This identifier indicates the subject category. "access-subject" is the default value.

`urn:oasis:names:tc:xacml:1.0:subject:subject-id-qualifier`

This identifier indicates the security domain of the subject. It identifies the administrator and policy that manages the name-space in which the subject id is administered.

`urn:oasis:names:tc:xacml:1.0:subject:key-info`

This identifier indicates a public key used to confirm the subject's identity.

`urn:oasis:names:tc:xacml:1.0:subject:authentication-time`

This identifier indicates the time at which the subject was authenticated.

`urn:oasis:names:tc:xacml:1.0:subject:authn-locality:authentication-method`

This identifier indicates the method used to authenticate the subject.

urn:oasis:names:tc:xacml:1.0:subject:request-time

This identifier indicates the time at which the subject initiated the access request, according to the PEP.

urn:oasis:names:tc:xacml:1.0:subject:session-start-time

This identifier indicates the time at which the subject's current session began, according to the PEP.

The following identifiers indicate the location where authentication credentials were activated. They are intended to support the corresponding entities from the SAML authentication statement.

urn:oasis:names:tc:xacml:1.0:subject:authn-locality:ip-address

This identifier indicates that the location is expressed as an IP address.

The corresponding attribute SHALL be of data-type "<http://www.w3.org/2001/XMLSchema#string>".

urn:oasis:names:tc:xacml:1.0:subject:authn-locality:dns-name

This identifier indicates that the location is expressed as a DNS name.

The corresponding attribute SHALL be of data-type "<http://www.w3.org/2001/XMLSchema#string>".

Where a suitable attribute is already defined in LDAP [LDAP-1, LDAP-2], the XACML identifier SHALL be formed by adding the attribute name to the URI of the LDAP specification. For example, the attribute name for the userPassword defined in the RFC 2256 SHALL be:

<http://www.ietf.org/rfc/rfc2256.txt#userPassword>

[B.6. Resource attributes]

urn:oasis:names:tc:xacml:1.0:resource:resource-id

These identifiers indicate attributes of the resource. The corresponding attributes MAY appear in the <Resource> element of the request context and be accessed by means of a <ResourceAttributeDesignator> element, or by an <AttributeSelector> element that points into the <Resource> element of the request context. This attribute identifies the resource to which access is requested. If an <xacml-context:ResourceContent> element is provided, then the resource to which access is requested SHALL be all or a portion of the resource supplied in the <xacml-context:ResourceContent> element.

urn:oasis:names:tc:xacml:2.0:resource:target-namespace

This attribute identifies the namespace of the top element of the contents of the <xacml-context:ResourceContent> element. In the case where the resource content is supplied in the request context and the resource namespace is defined in the resource, the PDP SHALL confirm that the namespace defined by this attribute is the same as that defined in the resource. The type of the corresponding attribute SHALL be "<http://www.w3.org/2001/XMLSchema#anyURI>".

[B.7. Action attributes]

These identifiers indicate attributes of the action being requested. When used, they SHALL appear within the <Action> element of the request context. They SHALL be accessed by means of an <ActionAttributeDesignator> element, or an <AttributeSelector> element that points into the <Action> element of the request context.

urn:oasis:names:tc:xacml:1.0:action:action-id

This attribute identifies the action for which access is requested.

Where the action is implicit, the value of the action-id attribute SHALL be

urn:oasis:names:tc:xacml:1.0:action:implied-action

[B.8. Environment attributes]

These identifiers indicate attributes of the environment within which the decision request is to be evaluated. When used in the decision request, they SHALL appear in the <Environment> element of the request context. They SHALL be accessed by means of an <EnvironmentAttributeDesignator> element, or an <AttributeSelector> element that points into the <Environment> element of the request context.

urn:oasis:names:tc:xacml:1.0:environment:current-time

urn:oasis:names:tc:xacml:1.0:environment:current-date

urn:oasis:names:tc:xacml:1.0:environment:current-dateTime

[B.9. Status codes]

The following status code values are defined.

urn:oasis:names:tc:xacml:1.0:status:ok

urn:oasis:names:tc:xacml:1.0:status:missing-attribute

urn:oasis:names:tc:xacml:1.0:status:syntax-error

urn:oasis:names:tc:xacml:1.0:status:processing-error

[B.10. Combining algorithms]

The deny-overrides rule-combining algorithm has the following value for the ruleCombiningAlgId attribute:

urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:deny-overrides

urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:deny-overrides

urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:permit-overrides

urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:permit-overrides

urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:first-applicable

urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:first-applicable

urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:only-one-applicable

urn:oasis:names:tc:xacml:1.1:rule-combining-algorithm:ordered-deny-overrides

urn:oasis:names:tc:xacml:1.1:policy-combining-algorithm:ordered-deny-overrides

urn:oasis:names:tc:xacml:1.1:rule-combining-algorithm:ordered-permit-overrides

urn:oasis:names:tc:xacml:1.1:policy-combining-algorithm:ordered-permit-overrides

Appendix D - XACML Authorisation data-flow model

Source: eXtensible Access Control Markup Language (XACML) Version 2.0. OASIS Standard, 1 Feb 2005. [Online]: http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf

The major actors in the XACML domain are shown in the data-flow diagram of **Error! Reference source not found..**

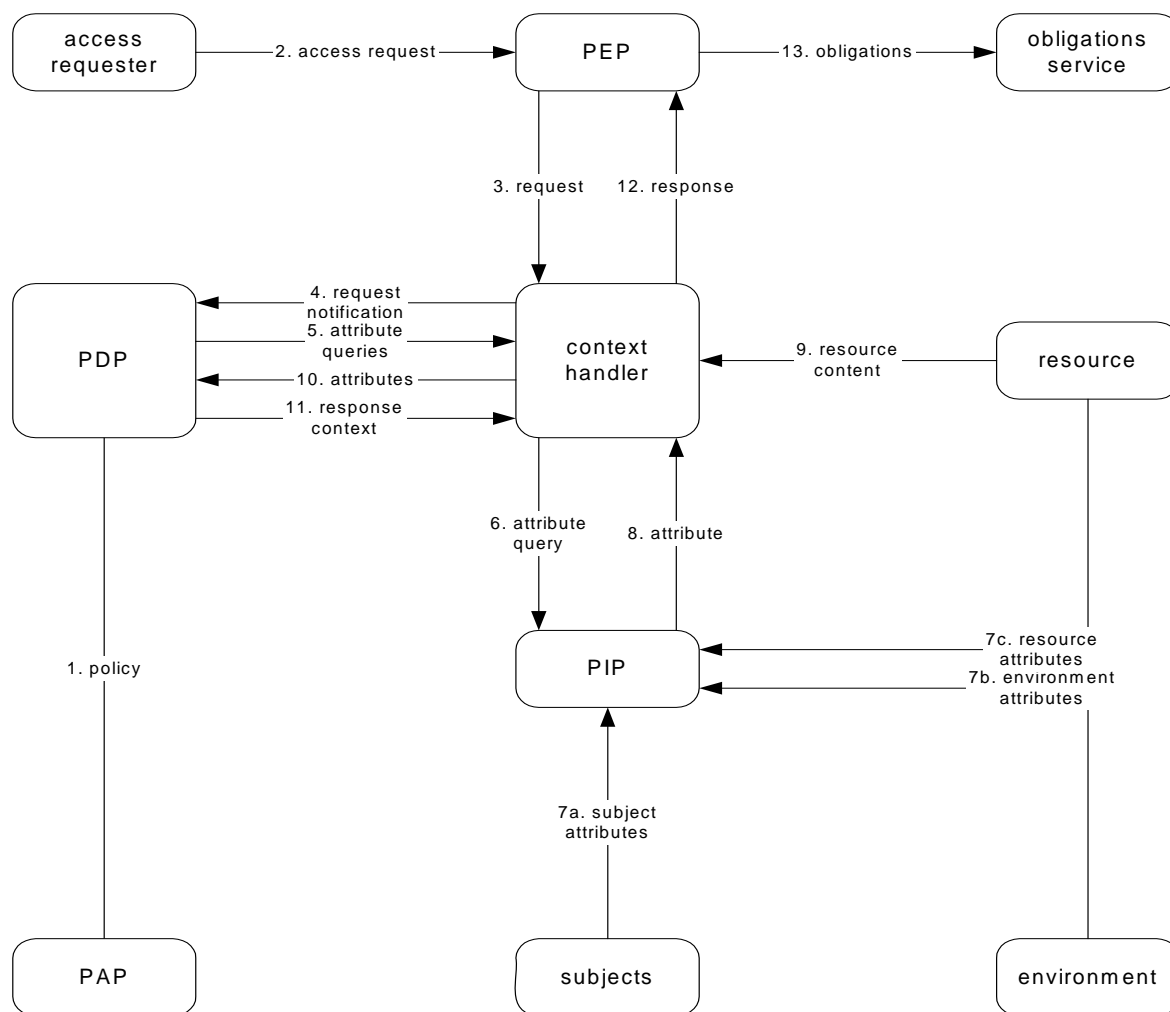


Figure 1 - Data-flow diagram

Note: some of the data-flows shown in the diagram may be facilitated by a repository. For instance, the communications between the **context** handler and the **PIP** or the communications between the **PDP** and the **PAP** may be facilitated by a repository. The XACML specification is not intended to place restrictions on the location of any such repository, or indeed to prescribe a particular communication protocol for any of the data-flows.

The model operates by the following steps.

1. **PAPs** write **policies** and **policy sets** and make them available to the **PDP**. These **policies** or **policy sets** represent the complete policy for a specified **target**.
2. The access requester sends a request for access to the **PEP**.
3. The **PEP** sends the request for **access** to the **context handler** in its native request format, optionally including **attributes** of the **subjects**, **resource**, **action** and **environment**.

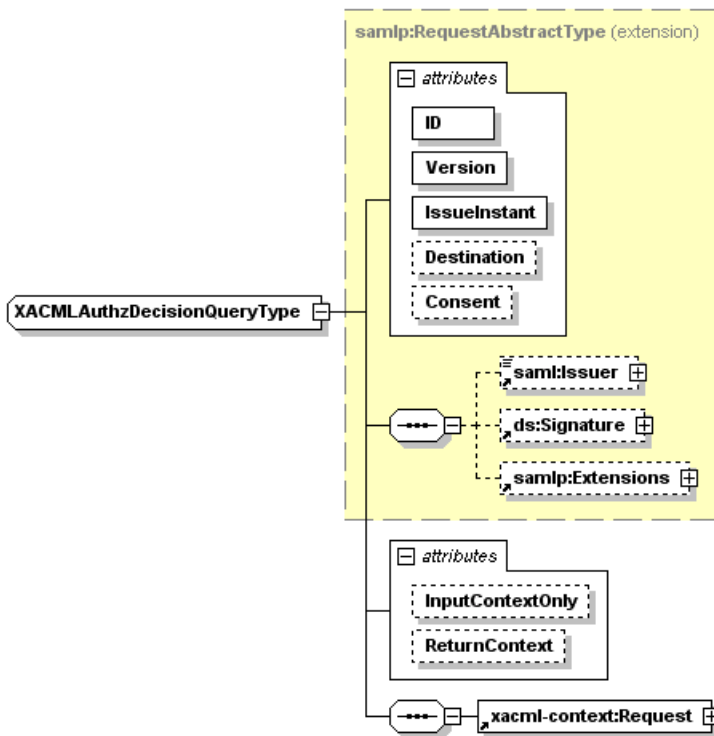
4. The **context handler** constructs an XACML request **context** and sends it to the **PDP**.
5. The **PDP** requests any additional **subject, resource, action** and **environment attributes** from the **context handler**.
6. The context handler requests the attributes from a **PIP**.
7. The **PIP** obtains the requested **attributes**.
8. The **PIP** returns the requested **attributes** to the **context handler**.
9. Optionally, the **context handler** includes the **resource** in the **context**.
10. The **context handler** sends the requested **attributes** and (optionally) the **resource** to the **PDP**. The **PDP** evaluates the **policy**.
11. The **PDP** returns the response **context** (including the **authorization decision**) to the **context handler**.
12. The **context handler** translates the response **context** to the native response format of the **PEP**. The **context handler** returns the response to the **PEP**.
13. The **PEP** fulfills the **obligations**.
14. (Not shown) If **access** is permitted, then the **PEP** permits **access** to the **resource**; otherwise, it denies **access**.

Appendix E - SAML2.0 profile of XACML overview

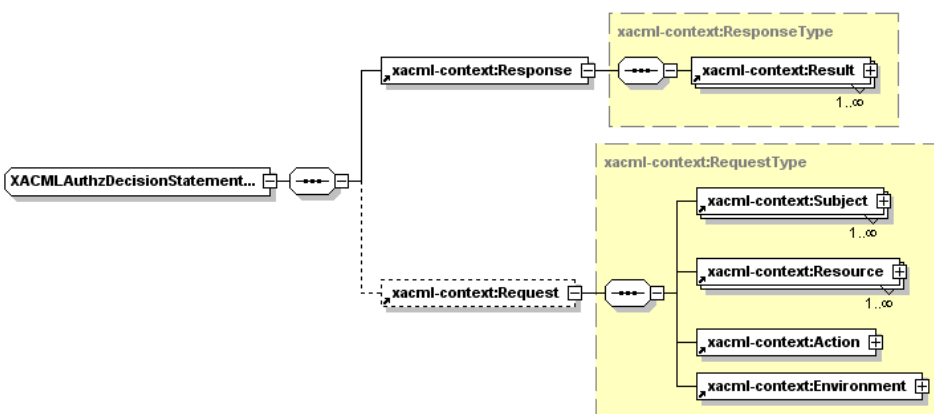
The SAML2.0 profile of XACML defines the queries and assertions to support XACML based AuthZ services.

The XACMLAuthzDecisionQuery and XACMLPolicyQuery provide extension to the SAML protocol. The XACMLAuthzDecisionStatement and XACMLPolicyStatement provide extension to the SAML assertions.

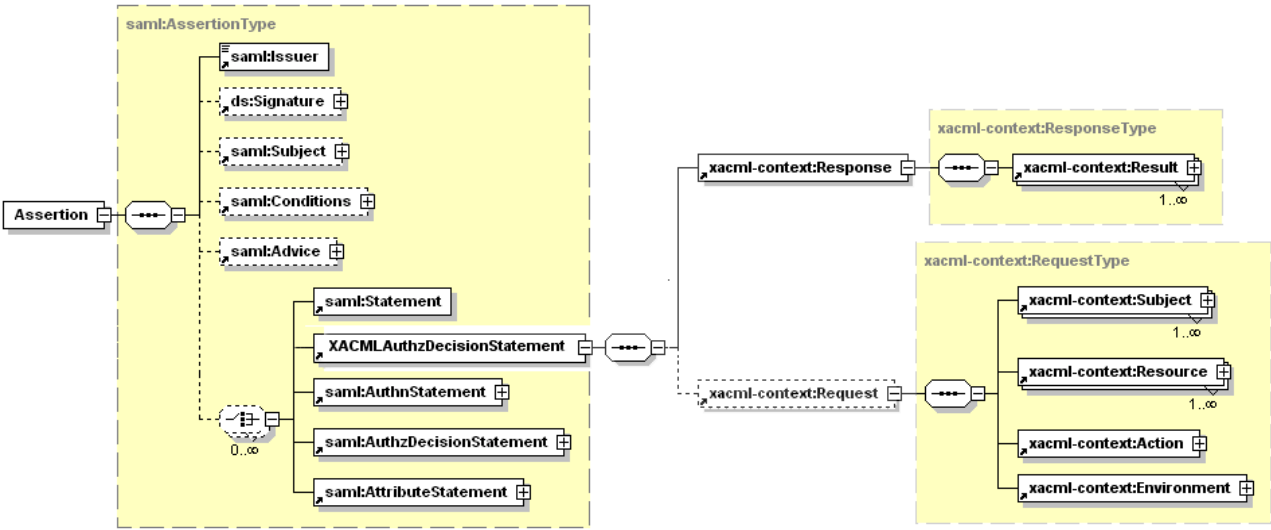
The XACMLAuthzDecisionQuery is introduced as additional query type for the SAML2.0 protocol. In contrary to the basic SAML2.0 queries, the XACMLAuthzDecisionQuery doesn't contain the Subject element but used as container for the xacml-context:Request message.



The XACMLAuthzDecisionStatement provides a container for XACML Request and Response messages that actually hold all necessary information about the AuthZ decision in a native XACML format.



It is important to note that XACMLAuthzDecisionStatement is not the assertion and need to be folded into the SAML assertion as shown below.



Appendix F - Policy on Multi-User Pilot Jobs

Excerpt from the JSP policy document - <https://edms.cern.ch/document/855383/1>

A multi-user pilot job, hereafter referred to simply as a pilot job, is a Grid job owned and submitted by one member of a Virtual Organisation (VO) which during execution at a Site pulls down and executes workload, hereafter called a user job, owned and submitted by a different member of the VO or multiple user jobs owned and submitted by multiple different members of the VO.

By submitting such a pilot job to the Grid, the VO and the owner of the pilot job agree to the conditions laid down in this document and other referenced documents, which may be revised from time to time.

- 1. Pilot jobs are only acceptable from VOs whose trust relationships with the Grid and/or Site include authorisation to use them.*
- 2. Pilot jobs must be owned and submitted by one of a limited number of authorised and registered members of the VO. The VO is responsible for implementing a process for authorising pilot jobs owners and ensuring that they accept the conditions laid down here. The pilot job owners are held personally responsible by the Grid and by the Site for the safe and secure operation of the pilot job and its associated user job(s).*
- 3. The pilot job must only execute user jobs belonging to registered and authorised members of the VO.*
- 4. The pilot job framework must meet the fine-grained monitoring and control requirements defined in the Grid Security Logging and Traceability policy. Glxexec in identity switching-mode is one implementation that meets these needs.*
- 5. The pilot job must use the approved system utility to map the application and data files to the actual owner of the workload and interface to local Site authorization, audit and accounting services. The owner of the user job is liable for all actions of that user job.*
- 6. The pilot job must respect the result of any local authorisation and/or policy decisions, e.g. blocking the running of the user job.*
- 7. The pilot job must not attempt to circumvent job accounting or limits placed on system resources by the batch system, for example the execution of more parallel jobs than allowed.*
- 8. A pilot job must protect all local data files created during the execution of one user job from the next user job.*
- 9. When fetching a user workload and credentials into the worker node, the pilot job must use means at least as secure as the original pilot job submission process.*
- 10. The Grid reserves the right to terminate any pilot jobs that appear to be operating beyond their authorisation and/or are not in compliance with this policy. Other possible consequences include blacklisting of users or the VO as a whole.*

11. *The VO and/or pilot job owner must produce and keep audit logs and must assist Grid Security Operations in security incident response.*

12. *The VO must make a description of the architecture, the security model and the source code of their pilot job system available to Grid Security Operations and/or Sites on request.*

This policy shall be signed for agreement by each of the authorised Pilot Jobs owners.

History of changes

- 1) Draft version 0.4 (future) – incorporate agreed namespace for AuthZ interoperability and provides reference to conformance test for EGEE-OSG-GT use cases for Obligations handling
- 2) Draft version 0.3 (April 29, 2008) – added more information on XACML and XACML-SAML profile for developers in Appendices
- 3) Draft version 0.2 (December 21, 2007) – updated after JRA1-AH meeting
- 4) Draft version 0.1 (October 18, 2007) – initial version presented for the discussion at the EGEE JRA1-AH meeting on October 24-26, 2007 at CERN
(http://indico.cern.ch/conferenceOtherViews.py?confId=17162&view=egee_meeting&showDate=all&showSession=all&detailLevel=contribution)