# Extension to the XACML AuthZ Interoperability Profile

Brian Bockelman*[1], Dave Dykstra†[2], David Groep‡[3], and Mischa Sallé§[3]

[1]University of Lincoln-Nebraska,
[2]Fermilab, Batavia, IL, USA
[3]FOM-Nikhef, Amsterdam, The Netherlands

November 12, 2014

**Abstract**

This document describes additions and clarifications to the XACML Grid Authorization Interoperability profile[2]. It is not intended as a replacement of the old profile, but as addition to it.

## 1 Introduction

The XACML Authorization Interoperability profile[2] was developed a number of years ago in a collaborative effort of OSG, EGEE and Globus, in order to agree on a common set of obligations and attributes to be used in the Grid authorisation infrastructure. It is currently used both by OSG and partly by EGI (as successor of EGEE) and has resulted among other things in the use of the same client software on both sites of the Atlantic.

Now that it has been successfully used for a number of years, a few shortcomings have come to light, which warrant extensions and adaptations. These changes will be described in the different subsections of Section 3. In Section 4 we will give their respective motivations.

A further motivation for adapting the profile could come from the Argus framework, the authorisation framework used by the majority of the European sites. Due to similar considerations as those expressed here, in particular the lack of certain obligations, the Argus collaboration has introduced a different profile[3]. By extending the interoperability profile to cover such obligations, we open the way to at some point unify the two profiles.

## 2 Notational conventions

The namespace of the profile is unchanged:

- Obligations have full ID
  *http://authz-interop.org/xacml/obligation/≪ObligationID≫*
- Attributes have full ID
  *http://authz-interop.org/xacml/attribute/≪AttributeID≫*

We will mostly use only the shortened IDs in this document. Further definitions:

*bbockelm@cse.unl.edu
†dwd@fnal.gov
‡davidg@nikhef.nl
§msalle@nikhef.nl

| | |
|---|---|
| UID: | User Identity |
| GID: | Group Identity |
| pGID: | Primary Group Identity |
| sGID: | Secondary Group Identity |
| DN: | Distinguished Name |
| EEC: | End Entity Certificate |
| VOMS: | Virtual Organisation Membership Service |
| AC: | Attribute Certificate |
| FQAN: | Fully Qualified Attribute Name |
| FQDN: | Fully Qualified Domain Name |

# 3 Extensions and adaptations

This section describes the different changes with respect to the previous interoperability profile. The rationale for the different changes is described in corresponding subsections of Section 4.

## 3.1 Full specification of *username* obligation

The specification of the obligation *username* (paragraph 7.5, reference [2]) only describes that it should set the username as given by the attribute *username*. The profile should be amended to read that the client sets the full account, according to `getpwent()` information [4] for the given username, i.e. UID, pGID and optionally sGIDs.

## 3.2 New *account* obligation

An obligation is required that can explicitly set primary or secondary groupnames, i.e. based on groupname instead of GID. To provide this, the profile should be extended with a new obligation *account* with attributes *username*, *primary-groupname* and *secondary-groupnames*. Each attribute is optional. The server may send the obligation without any attributes, in which case the client must verify that it has support for the obligation and fail if it has not. See also Section 3.5

**ID:** *account*

**Full Obligation ID:** *http://authz-interop.org/xacml/obligation/account*

**Attributes:**

**ID** : *username*
**Description:** username of the resulting account.
**Full Attribute ID:** *http://authz-interop.org/xacml/attribute/username*
**Type:** string
**Multiplicity:** 0 . . . 1

**ID** : *primary-groupname*
**Description:** primary groupname of the resulting account.
**Full Attribute ID:** *http://authz-interop.org/xacml/attribute/primary-groupname*
**Type:** string
**Multiplicity:** 0 . . . 1

**ID** : *secondary-groupname*
**Description:** secondary groupname of the resulting account.
**Full Attribute ID:** *http://authz-interop.org/xacml/attribute/secondary-groupname*
**Type:** string
**Multiplicity:** 0 . . . N

## 3.3 Behaviour multiple primary group (and username) attributes

The profile should be extended to enforce the same behaviour for primary GIDs as it prescribes for multiple UIDs (paragraphs 7.3 and 7.5, reference [2]):

1. Each obligation can contain at most one pGID setting attribute
2. If multiple obligations set a primary GID, either directly or indirectly, all resulting pGIDs should be identical.

## 3.4 Dependencies of the *secondary-gids* obligation

The old specification of the *secondary-gids* obligation stated this obligation needs the *uidgid* obligation. This requirement is removed: the server may send back a response containing an incomplete mapping (or no mapping at all). It is up to the client to determine whether this is a failure or not, and the client may obtain the actual mapping via other means.

Additionally, the server may also send back a combination of the *secondary-gids* together with another obligation than the *uidgid* obligation to produce a complete mapping.

## 3.5 Obligation attributes and their multiplicities

The multiplicity of the attributes for the different obligations shall be the following:

**ObligationID:** *username*[1]

**AttributeID:** *username*
**Multiplicity:** 0 . . . 1

**ObligationID:** *uidgid*[2]

**AttributeID:** *posix-uid*
**Multiplicity:** 0 . . . 1

**AttributeID:** *posix-gid*
**Multiplicity:** 0 . . . 1

**ObligationID:** *secondary-gids*[3]

**AttributeID:** *posix-gid*
**Multiplicity:** 0 . . . N

**ObligationID:** *account*

**AttributeID:** *username*
**Multiplicity:** 0 . . . 1
**AttributeID:** *primary-groupname*
**Multiplicity:** 0 . . . 1
**AttributeID:** *secondary-groupname*
**Multiplicity:** 0 . . . N

Each obligation may be send by the server without any attributes, in which case the client must verify that it has support for that obligation and fail if it has not.

## 3.6 Verification of subject attributes on client side

By optionally setting an issuer element in a subject attribute in the request (see paragraph 6.7 in reference [1]), the client can inform the server about the reliability of the corresponding attribute. We distinguish the following cases:

1. the client has no knowledge about the reliability of the attribute, i.e. it may or it may not be verified. In this case the client must not specify an issuer element.
2. the client can reliably state that the attribute is *not* verified. In this case the client should provide the issuer element with the special value *http://authz-interop.org/xacml/issuer/none*.

[1] §7.5, ref. [2]
[2] §7.3, ref. [2]
[3] §7.4, ref. [2]

3. the client can reliably state that the attribute *is* verified. In this case the client should provide the issuer element with a value depending on the type of attribute:
   (a) for appropriate non-VOMS attributes extracted from the proxy certificate, the value must be set to the issuer-DN of the EEC of the proxy chain, i.e. to the *subject-x509-issuer* subject attribute, see paragraph 6.1.4 in reference [2].
   (b) for appropriate VOMS attributes extracted from the VOMS AC inside the proxy certificate, the value must be set to the DN of the VOMS service that signed the corresponding AC, i.e. to the *voms-signing-subject* subject attribute, see paragraph 6.1.6 in reference [2].

   The full list of subject attributes with their respective issuers is given in the following table:

| **Attribute** | **Issuer element** |
|---|---|
| *subject-x509-id* | *subject-x509-issuer* |
| *subject-x509-issuer* | issuer-DN of CA certificate |
| *validity-not-before* | *subject-x509-issuer* |
| *validity-not-after* | *subject-x509-issuer* |
| *certificate-serial-number* | *subject-x509-issuer* |
| *ca-serial-number* | issuer-DN of CA certificate |
| *ca-policy-oid* | *subject-x509-issuer* |
| *cert-chain* | *subject-x509-issuer* |
| | |
| *vo* | *voms-signing-subject* |
| *voms-signing-subject* | *voms-signing-issuer* |
| *voms-signing-issuer* | issuer-DN of VOMS signing CA |
| *voms-fqan* | *voms-signing-subject* |
| *voms-primary-fqan* | *voms-signing-subject* |
| *voms-dns-port* | *voms-signing-subject* |
| | |
| *subject-condor-canonical-name-id* | FQDN or host certificate subject-DN of the service |

### 3.7 Extra requirement for backwards compatibility

The interoperability profile explicitly gives a list of best practise recommendations in paragraph 4.2. Although implied it does not mention explicitly what behaviour is expected from a client if it receives a known obligation with unknown attributes. The behaviour should be that the client fails. Hence it must be strongly discouraged to add extra attributes to existing obligations. Instead, when existing obligations cannot fulfil a use case, new obligations must be created.

## 4 Rationale

This section describes the rationale for the different changes described in Section 3. The different subsections in this section correspond to the changes described there.

### 4.1 Rationale: Full specification of *username* obligation

Both LCMAPS plugins (lcmaps-plugins-scas-client and lcmaps-plugins-c-pep) use an implementation that sets a complete account belonging to the username, according to `getpwent()` information [4]. Changing this behaviour would change the effect of the obligation in a backwards incompatible way. Note that the profile only supports passing supported obligationIDs to the server, not the corresponding attributeIDs, while clients typically fail on unrecognised attributes.

The new behaviour is also in line with 'classic' Globus behaviour, the Globus callout mechanism typically returns a single username, which is used to determine the complete account.

## 4.2 Rationale: New *account* obligation

The rationale for introducing a new obligation is motivated by the requirement to keep the username obligation implementation backwards compatible, see Subsection 4.1. Hence we cannot add additional primary and secondary groupname attributes to the username obligation.

Alternatively, introducing separate new obligations for the primary and secondary groupnames, analogously to the GID obligations, would have let to an asymmetry with respect to the username obligation: the username obligation would set the whole account, the groupname obligations only one or more groups.

The choice for a single new obligation with optional attributes leads to the greatest flexibility and the least amount of needed client code, since it can be handled with a single new obligation handler. Furthermore, it is very similar to the *http://glite.org/xacml/obligation/local-environment-map/posix* obligation in the Argus worker node profile (paragraph 3.5.1, ref. [3]).

## 4.3 Rationale: Behaviour multiple primary group attributes

The profile only described the behaviour in case multiple obligations set a UID, in which case either all of them must be identical, or the client should fail. The profile did not mention similar behaviour in case of multiple pGIDs, since it only contained a single obligation being capable of that, the uidgid obligation (even though in practise also the username obligation sets one). Since obligations are unordered in the XACML2 standard, there is a need to determine the behaviour. The new behaviour is the same as that for the UID.

## 4.4 Rationale: Dependencies of the *secondary-gids* obligation

For greater flexibility, any combination of credential mapping obligations is allowed as long as they are not conflicting.

## 4.5 Rationale: Obligation attributes and their multiplicities

It was unclear whether obligations were allowed to have no attributes, except for the *secondary-gids*, for which it was stated explicitly that the list of GIDs may be empty. Hence we formalise the full set of multiplicities. Changing the multiplicity for the attribute of the *secondary-gids* obligation would break current SCAS implementations, hence we keep the existing behaviour.

## 4.6 Rationale: Verification of subject attributes on client side

In the OSG scenario, the VOMS credentials are passed from the client to the server unverified. The user effectively passes a list of requested FQANs to the server (GUMS), and it is the task of the server to verify whether the user is allowed them.

In the European scenario, the client (gLExec) verifies the proxy, including its VOMS part. Hence the entire proxy chain is fully verified on client side, and the server (SCAS) can rely on the validity of the received credentials.

It is useful to have the ability to pass to the server who is responsible for the verification. This could speed-up GUMS performance (it does not need to verify them again if they are already verified) and provide feedback to the SCAS in case of a misconfiguration that would lead to reliance on unverified credentials.

A further motivation for adding issuer elements to the attributes comes from proxies containing multiple VOMS ACs. In those cases, having the issuer is the only way of telling which attributes belong to which VO.

### 4.6.1 Format and name-space

Although using RFC2253 [5] formatted x500name notation for DNs in the issuer elements would seem a more logical choice, this would constitute an incompatible change from the existing profile. Furthermore, obtaining RFC2253 formatted DNs from the current VOMS implementation is rather involved and would require changes in multiple libraries.

Concerning the name-space of the special none issuer, we have a preference to stay with the same name-space as the rest of the profile, i.e. *http://authz-interop.org/xacml*.

### 4.7 Rationale: Extra requirement for backwards compatibility

Since the profile provides all the necessary means to stay backwards compatible and still exchange sufficient information about understood obligations, via the *pep-oblig-supported* environment attribute (paragraph 6.5.2, ref. [2]), we do not need to provide a separate environment attribute providing the profile version, such as is used by the Argus worker node profile (paragraph 3.1.1, ref. [3]). The use of the supported obligations attribute is more flexible: it provides fully namespaced obligation IDs, allowing even for mixing different profiles.

## 5 Acknowledgements

This work is supported by SURFnet and ...

*Need acknowledgements*

## References

[1] Moses, T. (editor),
*eXtensible Access Control Markup Language (XACML) Version 2.0*, 2005,
`http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf`

[2] Ananthakrishnan, R. et al.,
*An XACML Attribute and Obligation Profile for Authorization Interoperability in Grids*,
2013 `https://www.ogf.org/documents/GFD.205.pdf`
2011 `http://cd-docdb.fnal.gov/cgi-bin/ShowDocument?docid=2952`

[3] La Joie, C. and Tschopp, V.,
*XACML Grid Worker NodeProfile, Version 1.0*, 2010,
`https://edms.cern.ch/document/1058175`

[4] The Open Group Base Specifications Issue 7, IEEE Std 1003.1, 2013 Edition,
*endpwent, getpwent, setpwent - user database functions*,
`http://pubs.opengroup.org/onlinepubs/9699919799/functions/endpwent.html`

[5] Wahl, M. et al.,
*Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished Names*, 1997
`http://tools.ietf.org/html/rfc2253`

*probably need more references, e.g. for GUMS, SCAS etc., RFC2253*