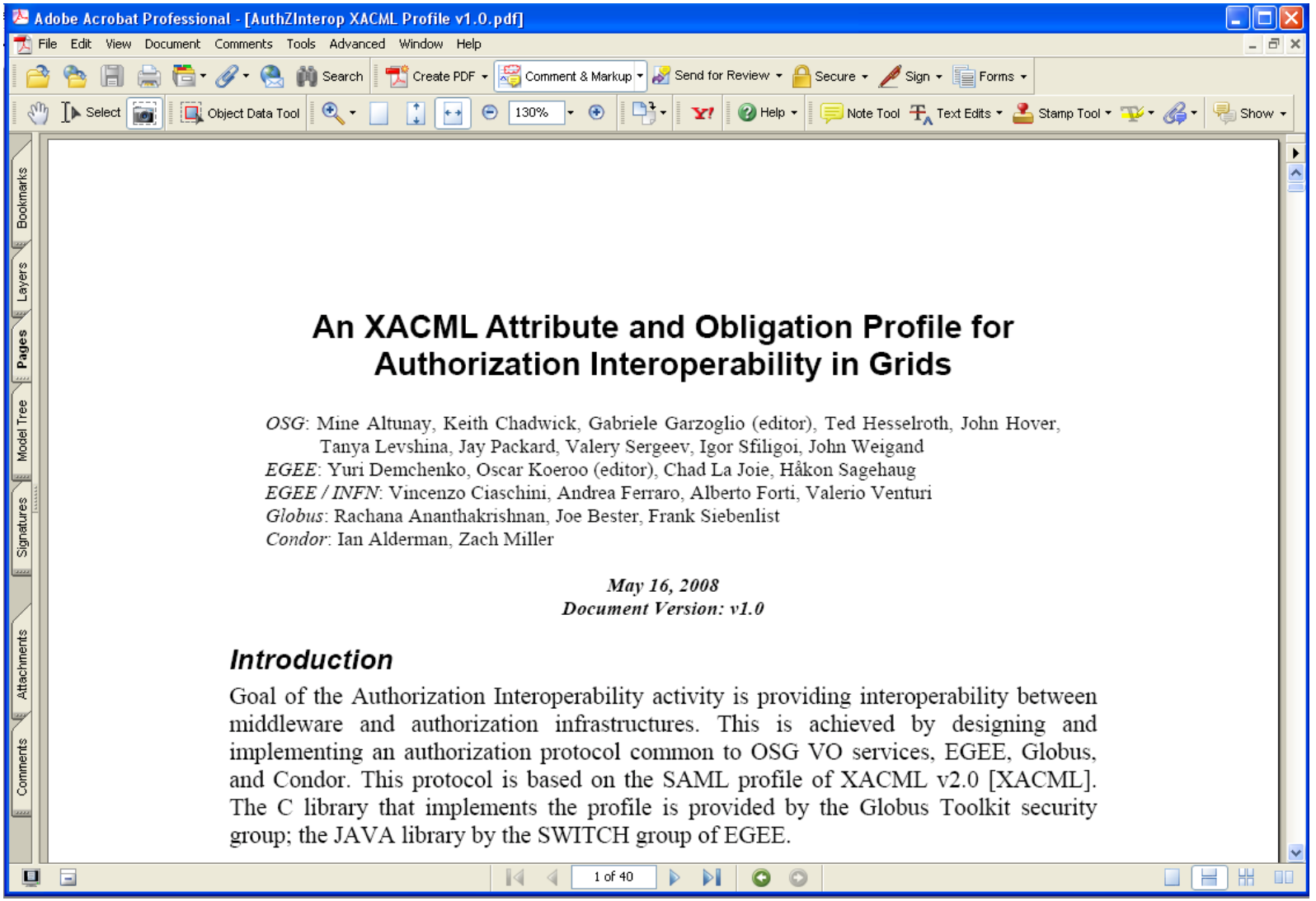
An XACML Attribute and Obligation Profile
for
Authorization Interoperability in Grids
(XACML-Grid Profile)

http://cd-docdb.fnal.gov/cgi-bin/ShowDocument?docid=2685

On behalf of Oscar Koeroo and Gabriele Garzoglio
Yuri Demchenko
SNE Group, University of Amsterdam

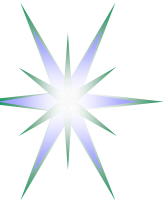
OGSA-AUTHZ WG, OGF 23
2 June 2008, Barcelona

Document View



Version 1.0 released on May 16, 2008

Result of wide cooperation between OSG, EGEE, Globus

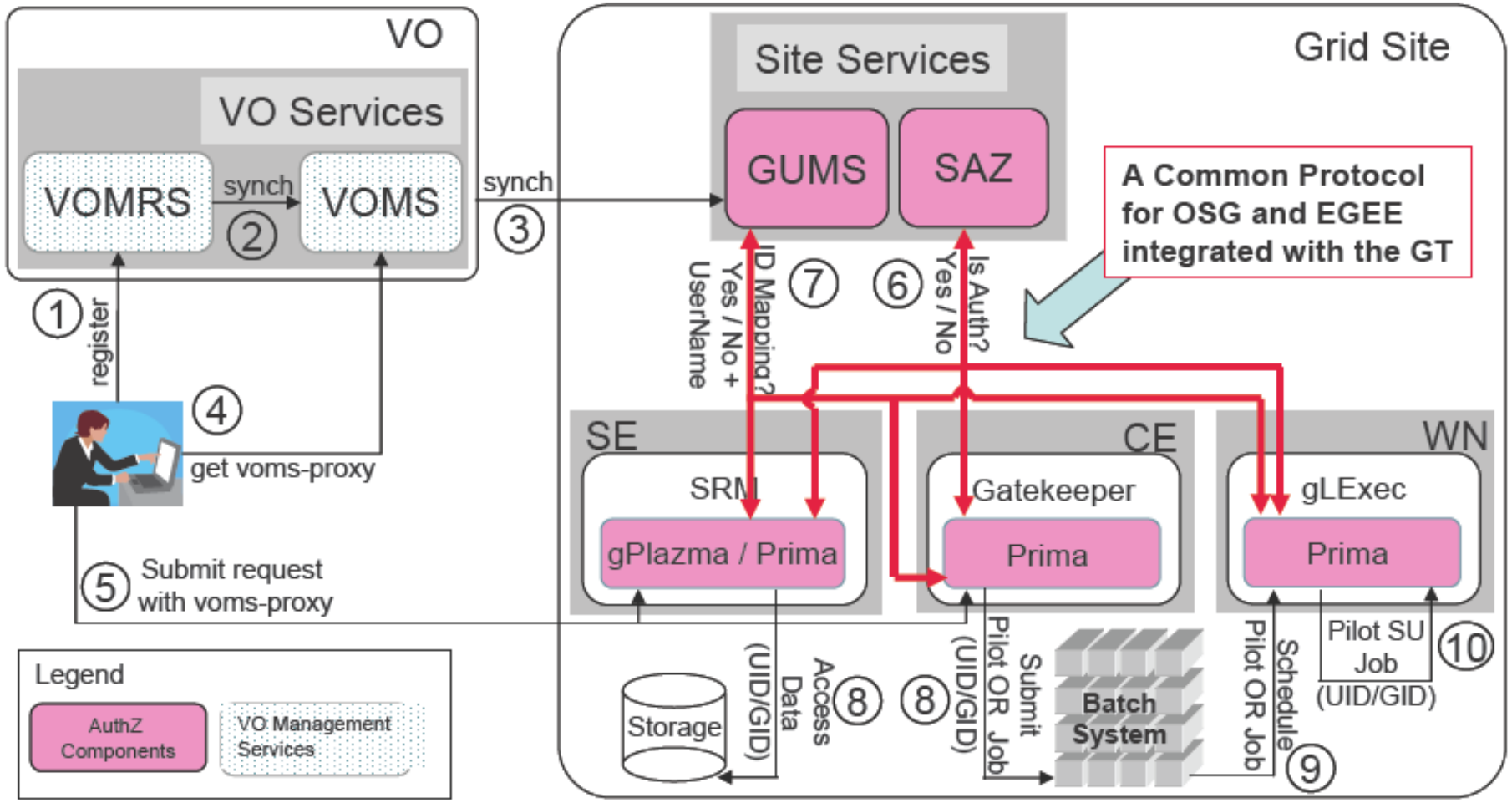


Outline

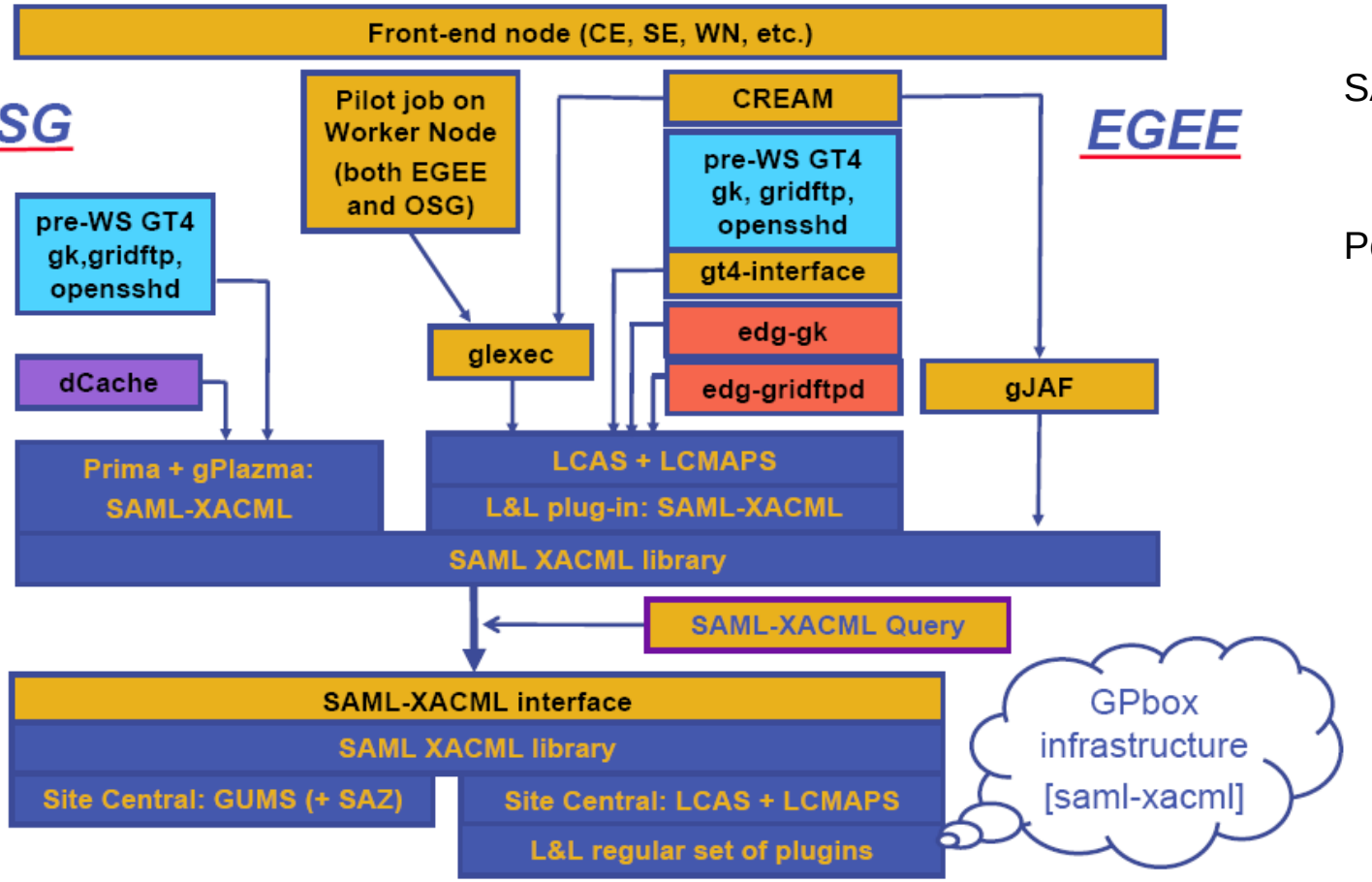
- Architecture view
- XACML Policy and XACML Request-Response
- Request attributes – Subject, Resource, Action, Environment
- Obligations and Obligations attributes
- Implementation

- Additional materials
  - Informal requirements
  - Obligations in Pilot Job handling
  - SAML-XACML Extension Library for OpenSAML2.0

Architecture - OSG view



Authorisation Interoperability – EGEE view



SAML-XACML profile as interoperability framework

Policy Obligation concept/mechanism identified as a solution to allow specific for Grid account mapping and other types of AuthZ decision enforcement (quota, path, priority)

XACML Policy format

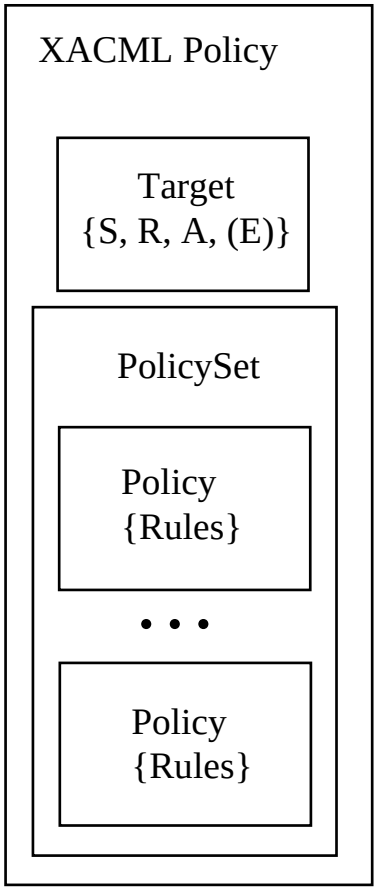
Policy consists of Policy Target and Rules

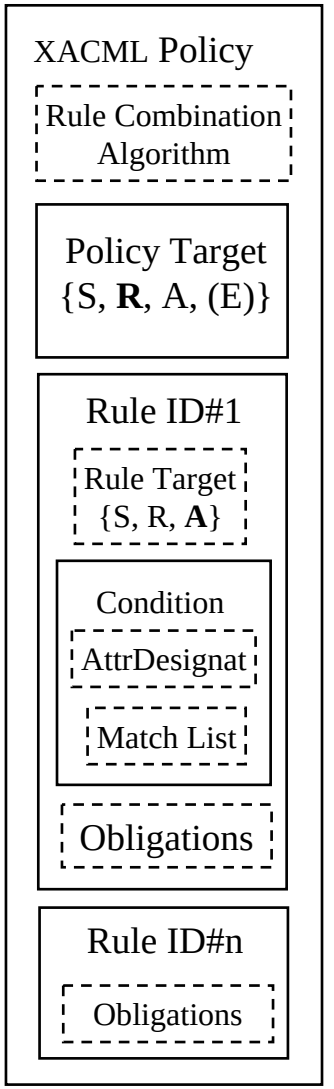
- Policy Target is defined for the tuple Subject-Resource-Action (-Environment)
- Policy Rule consists of Conditions and may contain Obligations

Obligations are a set of operations that must be performed by the ***PEP*** in conjunction with an ***authorization decision*** [XACML2.0]

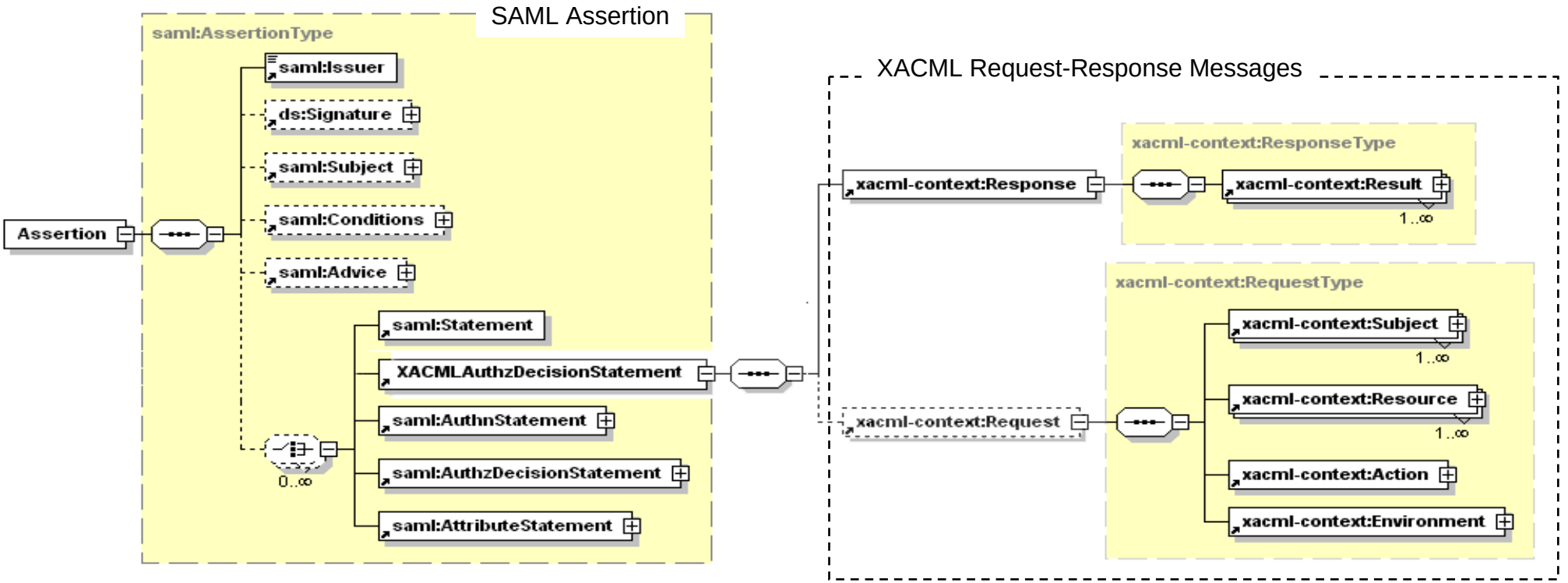
Obligations enforcement scenarios

- 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 accessed/consumed





SAML-XACML Request/Response messages



XACMLRequest (Resource, Subject, Action, Environment)

XACMLResponse (Result (ResourceId, Obligations?))

XACML Request-Response messages are enclosed into the SAML2.0 Assertion or SAML2.0 protocol messages

Namespace

Two options were discussed and evaluated - URN vs URL

- URL-style doesn't require centralized registration
- Can be established by registering the (relevant) domain name to ensure uniqueness

XACML-Grid uses registered namespace (owned by David Groep):

- http://authz-interop.org/

Root namespace prefix for all our message elements:

- http://authz-interop.org/xacml

Request Attribute Identifiers

Namespace prefix: *http://authz-interop.org/xacml*

Subject: *<ns-prefix>/subject/<subject-attr-name>*

Action: *<ns-prefix>/<action-attr-name>*

Resource: *<ns-prefix>/<resource-attr-name>*

Environment: *<ns-prefix>/environment/<env-type>*

Subject attributes (1)

Subject-id ⇒ Subject-X509-id

- String: OpenSSL oneline notation of the DN

Subject-X509-Issuer

- String: OpenSSL oneline notation of the Issuer DN

Subject-Condor-Canonical-Name-id

- String: “user@host[.domain]”

Subject-VO

- String: “gin.ggf.org”

VOMS-signing-subject

- String: OpenSSL oneline notation

VOMS-signing-issuer

- String: OpenSSL oneline notation

VOMS-FQAN

- String: “/gin.ggf.org/APAC/VO-Admin”

VOMS-Primary-FQAN

- String: “/gin.ggf.org/APAC/VO-Admin”

Subject attributes (2) - Optional

Certificate-Serial-Number

- Integer: 42

CA-serial-number

- Integer: 1

Subject End-Entity X509v3 Certificate Policies OID

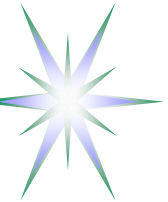
- String: “1.2.840.113612.5.2.4” (Robot Certificate)

Cert-Chain

- base64Binary: “*MIICbjCCAVagA……..*”

VOMS-dns-port

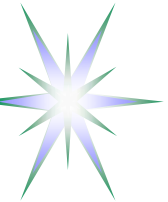
- String: “kuiken.nikhef.nl:15050”



Action attributes

Run-type: expressed as the ‘action-id’ (enumerated type)

- Queue
  - Requesting execution to a (remote) queue.
- Execute-Now
  - Requesting direct execution (remotely)
- Access (file)
  - Request for (generic) file access
- Resource Specification Language
  - RSL string



Resource attributes

Node-type: (enumerated type)

- CE (Computing Element)
  - Can also be the head-node or entry point to a cluster
- WN (Worker Node)
  - A node type that will process jobs, typically in a cluster
- SE (Storage Element)
  - (Logical) storage facility or specific storage node

Host DNS Name

- dns-host-name

Resource related attributes

- Resource X509 Service Certificate Subject
  - resource-x509-id
- Resource X509 Service Certificate Issuer
  - resource-x509-issuer

Environment attributes

PEP-PDP capability negotiation - Supported Obligations

- PEP sends to PDP a list of the supported obligations
- The PDP can choose to return an appropriate set of obligations from this list
- Allows upgradeability of the PEPs and PDPs independently by deploying new functionalities step by step

Pilot Job context

- To support pull-based job management model
- Policy statement example
  - “User access to the WM execution environment can be granted only if the pilot job belongs to the same VO as the user VO”
- Pilot job invoker identity
  - These attributes the the identity of the pilot job invoker

Obligations format and Id's

General format

Obligation = Apply (TargetAttribute, Operation (Variables))

Used in XACML-Grid profile (simplified)

Obligation = {AttributeAssignment (ObligationId, AttributeValue(AttributeId))}

ObligationId: *<ns-prefix>/obligation/<obligation-name>*

AttributeId: *<ns-prefix>/attributes/<obligation-attribute-name>*

Obligations (1)

UIDGID

- UID (integer): Unix User ID local to the PEP
- GID (integer): Unix Group ID local to the PEP

- Stakeholder: Common
- Must be consistent with: Username

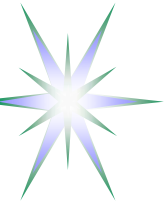
Multiple Secondary GIDs

- Multi recurrence
  - GID (integer): Unix Group ID local to the PEP
- Stakeholder: EGEE
- Needs obligation(s): UIDGID

Username

- Username (string): Unix username or account name local to the PEP.

- Stakeholder: VO Services Project
- Must be consistent with: UIDGID



Obligations (2)

AFSToken

- AFSToken (string) in base64: AFS Token passed as a string

- Stakeholder: EGEE
- Needs obligation(s): UIDGID

Obligations (3)

Path restriction (root-and-home-paths)

- RootPath (string): this parameter defines a sub-tree of the whole file system available at the PEP. The PEP should mount this sub-tree as the “root” mount point (‘/’) of the execution environment. This is an absolute path.
- HomePath (string): this parameter defines the path to home areas of the user accessing the PEP. This is a path relative to RootPath.

- Stakeholder: VO Services Project
- Needs obligation(s): UIDGID or Username

Storage Priority

- Priority (integer): an integer number that defines the priority to access storage resources.

- Stakeholder: VO Services Project
- Needs obligations: UIDGID or Username

Access permissions

- Access permission to a file that is requested
- Allowed values: “read-only”, “read-write”

Implementation

- In C – actually SAML-XACML front-end for LCAS/LCMAPS based Site Central AuthZ Services (SCAS)
  - The library is based on gSOAP – developed by Globus
  - C-based SCAS is operational and target June 2008 for release

- In Java
  - SAML2-XACML profile implemented as part of the recent OpenSAML2.0 library
  - Programming guidelines and examples
    - http://www.bccs.uib.no/~hakont/SAMLXACMLExtension/
  - Java-based SCAS is being developed as part of the Privilege project

XACML Obligations – Examples of expression for pool account mapping in Grid – Option 1 (used in XACML-Grid v1.0)

```
<!-- Obligations format option 1 (simple): UID, GID explicitly mentioned as
   separate XML elements inside AttributeAssignment element -->
<xacml:Obligations>
   <xacml:Obligation
        ObligationId=http://authz-interop.org/xacml/obligation/uidgid
        FulfillOn="Permit">
     <xacml:AttributeAssignment
          AttributeId=http://authz-interop.org/xacml/attribute/posix-uid
          DataType="http://www.w3.org/2001/XMLSchema#integer">
       2501</xacml:AttributeAssignment>
     <xacml:AttributeAssignment
          AttributeId=http://authz-interop.org/xacml/attribute/posix-gid
          DataType="http://www.w3.org/2001/XMLSchema#integer">
       2101</xacml:AttributeAssignment>
   </xacml:Obligation>
<xacml:Obligations>
```

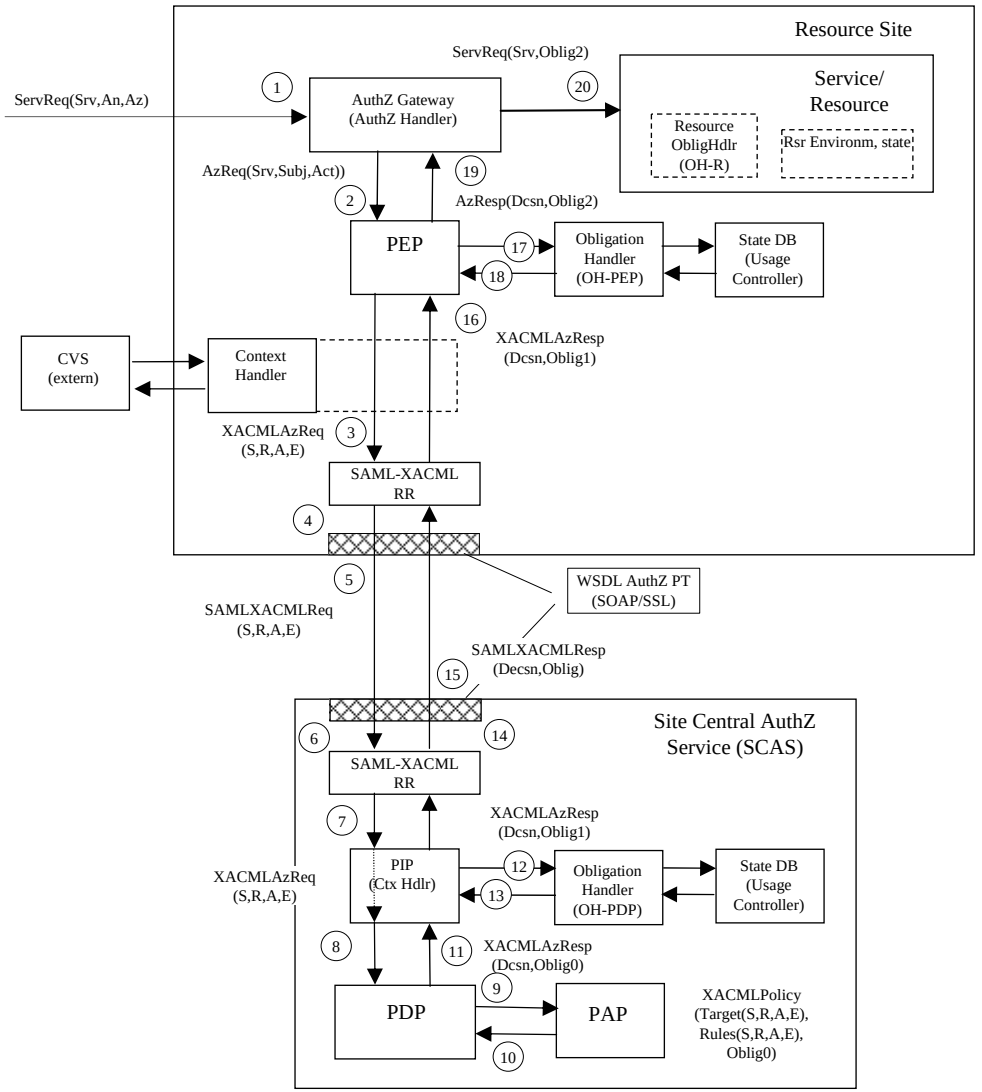
XACML Obligations – Examples of expression for pool account mapping in Grid – Option 2

```
<!-- Obligations format option 2: contains target attribute and what values to be assigned -->
<Obligations>
<Obligation ObligationId="http://authz-interop.org/xacml/obligation/map.poolaccount"
   FulfillOn="Permit">

<!-- This part specifies 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/value to which it should be mapped -->
<AttributeAssignment
     AttributeId="http://authz-interop.org/xacml/obligation/attribute/uidgid"
     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>
```

Future developments

- Conformance test for XACML-Grid profile and SAML-XACML library
- Obligations Handling API
- Obligations Handling model
  - Starting from closer look at obligation types and handling dataflow
- Extension for other related use case
  - E.g., Grid-enabled Network Resource Provisioning

Proposed Obligations Handling Reference Model



Generic AuthZ service model

PEP – Policy Enforcement Point

PDP – Policy Decision Point

PAP – Policy Authority Point

OH – Obligation Handler

CtxHandler – Context Handler

(S, R, A, E) – components of the AuthZ request
(Subject, Resource, Action, Environment)

Obligations Handling Stages

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

Obligation0 – (stateless or template)

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.

Obligation1 and Obligation 2

Obligations have been handled by Obligation handler at the SCAS/PDP side or at the PEP side, depending on implementation. Templates or instructions of the Obligation0 are replaced with the real attributes in Obligation1/2, e.g. in a form of “name-value” pair.

- The result of Obligations processing/enforcement is returned in a form of modified AuthzResponce (Obligation1) or global Resource environment changes
- 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. Obligation1 handling at the SCAS or PDP side allows stateful PDP/SCAS.

Obligation3

Final stage when an Obligation actually takes effect (Obligations “termination”). This is done by the Resource itself or by services managed/controlled by the Resource.

Additional Information

- Informal requirements
- Pilot Job submission process
- OpenSAML SAML-XACML Extension Library

Informal Requirements (Jun'07) - 1

The library should be usable outside of the Globus Toolkit framework

- However, the GT4 PEP are natively integrated

The library should support remote or local attribute validations

- The library should support sending signed assertions through the wire
- We will need to standardize the attribute names used in the assertion, to have a consistent semantics across implementations

The library should allow signing assertions with different certificates

- For example host cert, user cert, pilot admin cert, etc.

The library should be able to send some of the PEP context to the PDP

- For example: job description parameters, RSL, etc.
- The information could be passed to the PDP as a standardized XACML attribute.

The library should support arbitrary information from the PDP

- Using XACML Obligations…

Informal Requirements (Jun'07) - 2

Clients should be able to declare what obligations they can support

- We can use a standardized tag of the "environment" element
- Allows “upgradability” of the clients

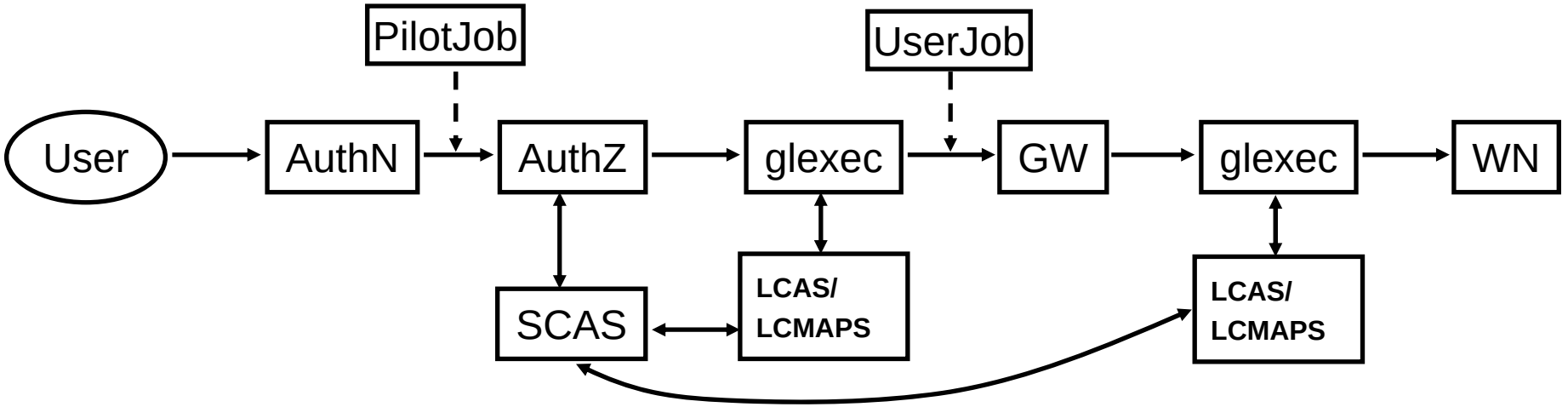
Handling of obligations should be implemented via external handlers

- Handlers will be associated to standardized obligation ids.

Interoperability profile should be used to generate test classes for the library (new)

- Check that our contexts (obligation structures, data types, etc.) do not break the wire representation

Obligations and Pilot Job use case



Introducing SCAS as external AuthZ service called from protected environment changes simple security model

- AuthN-AuthZ-glexec flow needs analysis
- Behind each (SCAS) policy should be clear operational model

SCAS is verified to be compatible with the XACML policy and PDP

- XACML uses pluggable security service model (i.e. called from major Service)
- glexec is a kind of gateway/border device

OpenSAML SAML-XACML extension library

Implements SAML2.0 profile of XACML2.0 Version 1 (with errata)

Builds upon the source of OpenSAML

Every XML-element/object in OpenSAML and the extension consists of

- An interface
- The implementation
- Builder for creating it
- Marshaller, Java->XML
- Unmarshaller, XML->Java

Supplementary code contains

- Helper class for making a XACML Request context from a SAML Assertion
- Examples/templates for creating SAML-XACML assertions and queries and extracting attributes and obligations