Using SAML and XACML for Complex Resource Provisioning in Grid based Applications

Yuri Demchenko, Leon Gommans, Cees de Laat

System and Network Engineering Group
University of Amsterdam

POLICY2007 Workshop
13-15 June 2007, Bologna
Outline

- General Complex Resource Provisioning (CRP) model
- gJAF components to support dynamic security context management
- AuthZ ticket format for extended AuthZ session management
- XACML Obligations – Implementation suggestions
- Future developments
- Additional materials
  - AuthZ service mechanisms and components
  - XACML policy examples

Background for this research

- EU funded Phosphorus Project “Lambda User Controlled Infrastructure for European Research” (EC Contract number 034115)
- EU funded EGEE (Enabling Grid for E-sciencE) Project (Reg. INFSO-RI- 508833)
- University of Amsterdam SNE Group ongoing research on GAAA-AuthZ – Generic Authentication, Authorization, Accounting (GAAA) AuthZ Framework
Complex Resource Provisioning (CRP)

Basic use cases for CRP

- **OLPP and Network on-demand provisioning**
- **Virtual Laboratory - Hierarchical and distributed resources and user attributes**
- **Grid Computing Resource – Virtualised, distributed and heterogeneous**

2 major stages/phases in CRP operation

- **Provisioning stage consisting of 4 basic steps**
  - Resource Lookup
  - Resource composition (including options)
  - Component resources reservation (reservation ID) including required AuthZ
  - Deployment

- **Access (to the resource) or consumption (of the consumable resource)**
  - Token Based Networking (TBN) reservation/AuthZ decision enforcement
CRP/OLPP infrastructure elements and basic sequences

- Provisioning sequences
  - Polling
  - Relay
  - Agent

TVS – Token Validation Service
DRAM – Dynamic Resource Allocation and Mngnt
PDP – Policy Decision Point
PEP – Policy Enforcement Point

User/Requestor

AAA
PDP

Agent
Agent Sequence

Relay Sequence

Polling Sequence

Resource Provisioning Session

Domain 1
NetworkElm/WSE

Domain 2
NetworkElm/WSE

Service/Applic
PEP/PDP

Application/AuthZ Session
Required AAA/Service plane functionality for CRP/OLPP

Authentication and Identity management
- Federated Identity and Federated Resource Access
- Attribute management (issue, validation, mapping, delegation)

Authorisation
- Multidomain AuthZ policy and/or decisions combination
- AuthZ session Management to convey AuthZ decision between domains

Trust management
- User and Resource based Federations (Shibboleth, NREN/GN2 AAI, VO)
  - Pre-established trust relations
- Dynamic trust relations based on dynamic (session based) security associations
  - We distinguish Resource access dynamic security and static data/resource security
- Initial trusted introduction
  - Trusted Computing Platform (TCG) based hardware rooted trust anchors
  - DNSSEC based VO certificates publishing
gJAF (gLite Java AuthZ Framework) Extensions to support extended Security Context management
GAAAPI components to support dynamic security context management

- Context Handler (CtxHandler) that provides a container for all Security Context information including initial Request context and policy Obligations
- TriagePDP to provide an initial evaluation of the request against AuthZ ticket stored in Cache
  - Used also for flexible AuthZ session management
- Ticket Authority (TickAuth) generates and validates AuthZ tickets or tokens on the requests from TriagePDP or ContextHandler
AuthZ Session management in gLite/GAAA-AuthZ

- AuthZ session is a part of the generic RBAC and GAAA-AuthZ functionality
- Session can be started only by an authorised Subject/Role
  - Session can be joined by other less privileged users
  - Session permissions/credentials can be delegated to (subordinate) subjects
- Session context includes Request/Decision information and may include any other environment or process data/information
  - AuthZ Session context is communicated in a form of extended AuthZ Assertion or AuthZ Ticket
  - SessionID is included into AuthzTicket together with other AuthZ Ctx information
  - Signed AuthzTicket is cached by the Resource PEP or PDP
- If session is terminated, cached AuthzTicket is deleted from Cache
  - Note: AuthzTicket revocation should be done globally for the AuthZ trust domain
AuthZ ticket/assertion for extended security context management – Data model (1) - Top elements

Required functionality to support multidomain provisioning scenarios
- Allows easy mapping to SAML and XACML related elements
- Allows multiple Attributes format (semantics, namespaces)
- Establish and maintain Trust relations between domains
  - Including Delegation
- Ensure Integrity of the AuthZ decision
  - Keeps AuthN/AuthZ context
  - Allow Obligated Decisions (e.g. XACML)
- Confidentiality
  - Creates a basis for user-controlled Secure session
AuthZ ticket Data model (2) - Mandatory elements

- TicketID attribute
- Decisions element and ResourceID attribute
- Conditions Element and validity attributes
- Extensible element ConditionAuthzSession
  - Any AuthZ session related data
AuthZ ticket Data model (3) – Subject and Delegation elements

- Subject element to keep AuthN security context and Subject Attributes
- Delegation element to allow permissions/AuthZ decision delegation to other Subjects or groups/community
AuthZ ticket main elements

**<Decision>** element - holds the PDP AuthZ decision bound to the requested resource or service expressed as the ResourceID attribute.

**<Conditions>** element - specifies the validity constrains for the ticket, including validity time and AuthZ session identification and additionally context

  - **<ConditionAuthzSession>** (extendable) - holds AuthZ session context

**<Subject>** complex element - contains all information related to the authenticated Subject who obtained permission to do the actions

  - **<Role>** - holds subject’s capabilities
  - **<SubjectConfirmationData>** - typically holds AuthN context
  - **<SubjectContext>** (extendable) - provides additional security or session related information, e.g. Subject’s VO, project, or federation.

**<Resources>/<Resource>** - contains resources list, access to which is granted by the ticket

**<Actions>/<Action>** complex element - contains actions which are permitted for the Subject or its delegates

**<Delegation>** element – defines who the permission and/or capability are delegated to: another DelegationSubjects or DelegationCommunity

  - attributes define restriction on type and depth of delegation

**<Obligations>/<Obligation>** element - holds obligations that PEP/Resource should perform in conjunction with the current PDP decision.
AuthZ ticket format (proprietary) for extended security context management

```xml
TicketID="cba06da9df148cf4200ef8f3e4fd2b3">
  <AAA:Decision ResourceID="http://resources.collaboratory.nl/Philips_XPS1">Permit</AAA:Decision>
  <!-- SAML mapping: <AuthorizationDecisionStatement Decision="*" Resource="*"> -->
  <AAA:Actions>
    <AAA:Action>cnl:actions:CtrlInstr</AAA:Action> <!-- SAML mapping: <Action> -->
    <AAA:Action>cnl:actions:CtrlExper</AAA:Action>
  </AAA:Actions>
  <AAA:Subject Id="subject">
    <AAA:SubjectID>WHO740@users.collaboratory.nl</AAA:SubjectID> <!-- SAML mapping: <Subject>/NameIdentifier> -->
    <AAA:SubjectConfirmationData>IGhAllIwa8YQomTgb9Eg9JRN184AggaDkOb5WW4U</AAA:SubjectConfirmationData> <!-- SAML mapping: EXTENDED <SubjectConfirmationData/> -->
    <AAA:Role>analyst</AAA:Role> <!-- SAML mapping: <Evidence>/<Assertion>/<AttributeStatement>/<Assertion>/<Attribute>/<AttributeValue> -->
    <AAA:SubjectContext>CNL2-XPS1-2005-02-02</AAA:SubjectContext> <!-- SAML mapping: <Evidence>/<Assertion>/<AttributeStatement>/<Assertion>/<Attribute>/<AttributeValue> -->
  </AAA:Subject>
  <AAA:Delegation MaxDelegationDepth="3" restriction="subjects">
    <!-- SAML mapping: LIMITED <AudienceRestrictionCondition> (SAML1.1), or <ProxyRestriction>/<Audience> (SAML2.0) -->
    <AAA:DelegationSubjects> <AAA:SubjectID>team-member-2</AAA:SubjectID> </AAA:DelegationSubjects>
  </AAA:Delegation>
    <!-- SAML mapping: <Conditions NotBefore="*" NotOnOrAfter="*"/> -->
    <AAA:ConditionAuthzSession PolicyRef="PolicyRef-GAAA-RBAC-test001" SessionID="JobXPS1-2006-001">
      <!-- SAML mapping: EXTENDED <SAMLConditionAuthzSession PolicyRef="*" SessionID="*"/> -->
      <AAA:SessionData>put-session-data-Ctx-here</AAA:SessionData> <!-- SAML EXTENDED: <SessionData/> -->
    </AAA:ConditionAuthzSession>
  </AAA:Conditions>
  <AAA:Obligations>
    <AAA:Obligation>put-policy-obligation(2)-here</AAA:Obligation> <!-- SAML EXTENDED: <Advice>/<PolicyObligation> -->
    <AAA:Obligation>put-policy-obligation(1)-here</AAA:Obligation>
  </AAA:Obligations>
</AAA:AuthzTicket>
```

AuthzToken example – 293 bytes

<AAA:AuthzToken TokenID="c24d2c7dba476041b7853e63689193ad">
  <AAA:TokenValue>
    0IZt9WsJT6an+tIxhhTPtiztDpZ+iynx7K7X2Cxd2iBwCUTQ0n61Sxzv81DKl1Wsq75IsHfusnm56zT3fhKU1zEUsob7p6oMLM7hb42+vjfvNeJu2roknhIDzruMrr6hMDsIfo3tURepu7QCTOsADm9IfX89Et55EkSE9oE9qBD8=
  </AAA:TokenValue>
</AAA:AuthzToken>

AuthzToken is constructed of the AuthzTicket TicketID and SignatureValue
AuthzToken use suggests caching AuthzTicket’s
XACML Obligations - Definition

Obligations semantics is not defined in the XACML policy language but left to bilateral agreement between a PAP and the PEP.

PEPs that conform with XACMLv2.0 are required to deny access unless they understand and can discharge all of the <Obligations> elements associated with the applicable policy.

Element <Obligations> / <Obligation>

- The <Obligation> element SHALL contain an identifier (in the form of URI) 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.

```xml
<x:s:element name="Obligation" type="xacml:ObligationType"/>
<x:s:complexType name="ObligationType">
  <x:s:sequence>
    <x:s:element ref="xacml:AttributeAssignment" minOccurs="0" maxOccurs="unbounded"/>
  </x:s:sequence>
  <x:s:attribute name="ObligationId" type="xs:anyURI" use="required"/>
  <x:s:attribute name="FulfillOn" type="xacml:EffectType" use="required"/>
</x:s:complexType>
```
XACML Obligations – Examples of expression for pool account mapping in Grid

```xml
<Obligations>
  <Obligation ObligationId="http://glite.egee.org/JRA1/Authz/XACML/obligation/map.poolaccount"
    FulfillOn="Permit">
    <AttributeAssignment AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:text"
      DataType="http://www.w3.org/2001/XMLSchema#string">
      <SubjectAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
        DataType="http://www.w3.org/2001/XMLSchema#string"/>
      <AttributeAssignment AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:mapto"
        DataType="http://www.w3.org/2001/XMLSchema#string">
        <UnixId DataType="http://www.w3.org/2001/XMLSchema#string">okoeroo</UnixId>
        <GroupPrimary DataType="http://www.w3.org/2001/XMLSchema#string">computergroup</GroupPrimary>
        <GroupSecondary DataType="http://www.w3.org/2001/XMLSchema#string">datagroup</GroupSecondary>
      </AttributeAssignment>
    </AttributeAssignment>
    <AttributeAssignment AttributeId="urn:oasis:names:tc:xacml:2.0:example:attribute:poolaccount"
      DataType="http://www.w3.org/2001/XMLSchema#string">
      <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"/>
    </AttributeAssignment>
  </Obligation>
</Obligations>
```
Obligation handling model proposed in the process of interoperability workshop between GT, OSG and EGEE

- ObligationId (of type URI) has to be mapped to a specific handler that is called by the PEP
- Obligation parameter values are passed to handler
- Handler returns True/False determines PEP’s Permit/Deny
- Possible standardization
  - Obligations semantics and interface for passing obligation parameters to the Handler
  - Add Chronicle {before, at, after} attribute to indicate when Obligations should be applied by PEP and Resource
Future developments

- Implement AuthZ session management using AuthZ ticket for popular AuthZ frameworks gJAF, GT-AuthZ, GAAA-AuthZ
  - Including delegation and complex and obligated policy decisions
  - Needs more discussion on Delegation use cases and scenarios
- Defining XACML policy profiles and mapping
  - For other legacy policy formats: gridmap, ACL, GACL
  - For different Resource models (hierarchical, ordered, mesh, etc.)
- Standardisation and other initiatives
  - Proposing AuthZ session management framework to OGSA-AUTHZ
  - Site Central AuthZ Service using SAML-XACML protocol and assertion
  - Defining Policy Repository Service (PRS) protocol
Additional information

- Generic AuthZ service components and mechanisms
- Simple XACML policy example for Collaborative application
Generic AuthZ Components and Mechanisms

- An "authorization" is a process by which a right or a permission is granted to an entity/subject to access a resource.

- AuthZ Service Components
  - Subject (ID, Attrs), Policy (Locality/Environment), Resource/Object (State)

- AuthZ service interoperation and compatibility
  - The same AuthZ decision on the same set of Subject attributes based on the same Resource state
    - May contain Conditions/Obligations implied by the Policy decision
  - Example 1: The same tour booked via different tourist offices (even if in different countries)

- Basic mechanisms for interoperability
  - Credentials/Attributes validation/mapping
  - AuthZ decision assertions or tickets (usually bound to AuthZ session)
  - Authority binding (to convey trust relations)
    - All credentials and policy should match authority/issuer
AuthZ Models and Frameworks

AuthZ service component models
- User/AuthZ session and attributes management – RBAC, ITU/ISO X.812 PMI, GAAA-AuthZ, AAI, Shibboleth
- Application integration – Interceptor/Axis model (gJAF, GT4-AuthZ, Acegi), generic AAA-API
- Policy type – BlackList, ACL, gridmap, XACML, PERMIS
- Credentials/Attributes – X.509 AC/VOMS, SAML, Shibboleth

Existing AuthZ frameworks
- EGEE gLite Java AuthZ Framework and Globus GT-AuthZ
- LCAS/LCMAPS
- PERMIS
- GAAA-AuthZ (by UvA)
- COPS (Common Open Policy Service) – RFC2748, RFC2753, RFC3761
- Acegi (for J2EE/Spring)
- Shibboleth, Liberty and A-Select based AAI
XACML Policy structure

XACML Policy format

RBAC/XACML Policy
- Target {S, R, A, (E)}
- PolicySet
- Policy {Rules}

XACML Policy
- Rule Combination Algorithm
- Policy Target {S, R, A, (E)}
- Rule ID#1
  - Rule Target {S, R, A}
  - Condition
    - AttrDesignat'
    - Match List
- Rule ID#n
CNL AuthZ policy: XACML Policy generation conventions

- Policy Target is defined for the Resource
- Policy combination algorithm is “ordered-deny-override” or “deny-override”
- Rule Target is defined for the Action and may include Environment checking
  - Rule’s Condition provides matching of roles which are allowed to perform the Action
- Access rules evaluation
  - Rules are expressed as permissions to perform an action against Subject role
  - Rule combination algorithm “permit-override”
  - Rules effect is “Permit”
- Subject and Credentials validation – is not supported by current XACML functionality
  - Credential Validation Service (CVS) – proposed GGF-AuthZ WG development
RBAC AuthZ policy: Resource, Actions, Subject, Roles

Actions (8)
- StartSession
- StopSession
- JoinSession
- ControlExperiment
- ControllInstrument
- ViewExperiment
- ViewArchive
- AdminTask

Roles (4)
- Analyst
- Customer
- Guest
- Administrator (CertifiedAnalyst)

Naming convention
- Resource - “http://resources.collaboratory.nl/Phillips_XPS1”
- Subject – “WHO740@users.collaboratory.nl”
- Roles - “role“ or “role@ExperimentID”
**Simple Access Control table**

<table>
<thead>
<tr>
<th>Roles</th>
<th>Analyst</th>
<th>Custom</th>
<th>Guest</th>
<th>Admin</th>
</tr>
</thead>
<tbody>
<tr>
<td>ContrExp</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ContrInstr</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>ViewExp</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>ViewArch</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>AdminTsk</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>StartSession</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>StopSession</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>JoinSession</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

See XACML policy example =>