Domain Based Access Control Model for Distributed Collaborative Applications

Yuri Demchenko
University of Amsterdam
demch@science.uva.nl
Leon Gommans
University of Amsterdam
lgommans@science.uva.nl

Cees de Laat
University of Amsterdam
delaat@science.uva.nl
Rene van Buuren
Telematica Instituut
Rene.vanBuuren@telin.nl

ABSTRACT

This paper describes the design and development of a flexible domain-based access control infrastructure for distributed Collaborative Environments. The paper proposes extensions to classical RBAC models to address typical problems and tasks in the distributed hierarchical resource organisation that came from the practical experience in developing industry oriented virtual laboratories infrastructure. The proposed extensions/solutions address the following problems: hierarchical resources policy administration, user roles/attributes management, dynamic security context and authorisation session management, and others. The paper provides implementation details on the use of XACML for fine-grained access control policy definition for domain based resources and roles organisation. Special attention is given to practical implementation of the authorisation session management as a key component of the distributed hierarchical access control infrastructure. The paper analyses the required functionality and suggests extensions to the major service-oriented access generic framework such as Acegi, Globus Toolkit Authorisation framework, and GAAA Authorisation framework in order to support complex resource organisation and collaboration scenarios in dynamic virtualised environments. The paper is based on experiences gained from the industry funded project Collaboratory.nl and other major Grid-based and Grid-oriented projects in collaborative applications and complex resource provisioning.

KEYWORDS: Open Collaborative Environment, Domain based security model, RBAC, SAML, XACML, Security context management.

1. INTRODUCTION

The process industry makes extensive use of advanced laboratory equipment, such as electron microscopes, equipment for surface analysis and mass spectrometers. Due to the high initial outlay and operational costs, and the expertise required to operate the equipment, laboratories tend not to have all this equipment in-house. On other side the equipment owners increasingly consider providing access to their resources to remote collaborative groups in a form of "virtual" laboratory (VL) that uses modern ICT infrastructures and Internet technologies. Such a VL can offer the same possibilities as a traditional laboratory, but also enables laboratory staff to utilise the equipment and expertise of third parties.

In industrial environment, management and assurance issues are paramount. They must be supported by corresponding security infrastructure that should provide secure instruments access and reliable service delivery, and also allow hierarchical administration and minimum privileges assignment.

Emerging Computer Grid and Web Services [1, 2] technologies provide a good basis/platform for building such an open Service-Oriented Collaborative Environment (SOCE) that allows dynamic association of resources and users into virtual organisations or laboratories. Such a virtualisation of resources and users can be created dynamically, based on experiment (or business) agreements and terminated once the experiment has been completed.

Grid middleware, been developed in the framework of large international projects such as EGEE[1] and Globus Alliance[2], provides a common communication/messaging infrastructure for all resources and services exposed as Grid services, and also allows for a uniform security configuration at the service container or messaging level.

[1] http://public.eu-egee.org/
[2] http://www.globus.org/

It has reached a production level of maturity, but it still remains primary focused on computational resources and tasks management.

This significantly simplifies development of SOCE applications and allows developers to focus on application-level logic such as providing advanced business process management and the delivery of complex domain-specific applications.

This paper describes our experiences when developing a flexible, customer-driven, security infrastructure for a SOCE. It proposes a domain-based security model to address specific and common problems when implementing Service Oriented Architecture (SOA) and Grid technologies in the industrial collaborative environment. It continues with further development of the Experiment-centric customer driven security model for Open Collaborative Environment proposed in [3, 4] and being developed in the framework of the Collaboratory.nl[3] project (CNL) that investigates how technologies for remote operation of laboratory equipment can be integrated with existing GroupWare for enhanced remote collaboration.

The paper is organized as follows. Section 2 describes typical VL organisations between cooperating enterprises provides justification for the domain-based security model (DM). It also provides suggestion on the practical implementation in SOCE and discusses benefits. Section 3 discusses what functionality is currently available in known Role-based Access Control (RBAC) implementations and identifies extensions to address specifics in controlling access to distributed hierarchical resources in DM. Section 4 goes deeper into authorisation service operation in a typical SOCE and identifies mechanisms to express and convey DM dynamic security context. Section 6 discusses what functionality should be added to existing authorisation frameworks to support domain related security context handling. Section 6 provides practical suggestions and an example of using XACML for policy expression in DM.

The proposed approach and solutions respond to both common and business domain-specific requirements of Collaboratory.nl, and are based on current experience in the EGEE project. The proposed approach and solutions can also be used for other use cases that require distributed, dynamically invoked and managed access control infrastructure using Grid and Web Services middleware.

2. DOMAIN BASED RESOURCE MANAGEMENT AND SECURITY SERVICES

VL provides a flexible framework for associating instruments, resources and users into distributed interactive cooperative/collaborative environment. However, committed to the VL resource still remain in the possession and under direct administration of their original owner enterprises.

The following administrative and security domains can be defined for user, resources, policy and trust management:

1) Facility – provides administrative/legal platform for all further operational associations; may define what kind of technologies, formats, credentials can be used.

2) Virtual Laboratory (VL) – similar to Virtual Organisation (VO) in Grids. VL can be created on the basis of the VL agreement that defines VL resources, common services (first of all, information/registry and security), administrative structure and VL administrator. Trust relations can be established via PKI and/or VL Certificate population.

3) Experiment/project is defined together with the VL resources allocation, members, task/goals, stages, and additionally workflow. It is perceived that experiment related context may change during its lifetime.

4) Experiment session that may include multiple Instrument sessions and Collaborative sessions that involves experiment members into interactions.

5) Collaborative session – user interactive session.

In the above provided classification domains are defined by common policy under single administration, common namespace and semantics, shared trust, etc. In this case, domain related security context may include: namespace aware names and ID's, policy references/ID's, trust anchors, authority references, and also dynamic/session related context.

As a dynamic entity or dynamic security association, Experiment session must be supported by the AuthZ session that is based on the positive AuthZ decision, with possible obligations and conditions. Dynamic character of the Experiment/AuthZ sessions allows also delegation of user rights/permissions and must be supported by AuthZ (session) tickets/credentials.

Proposed in [3, 4] and implemented at the CNL project Demo stage the Experiment-centric security model provides a good solution for centrally managed resources and user engaged into the Experiment but doesn't reflect real hierarchical and multidomain Resource management model in typical industrial environment. Without including these administrative levels into the resource and security management model, their management will remain manual work and will be resulted in slow adaptation of the working space, a high administrative overhead and overly complex management.

The Domain-based resource management model (DM) closer reflects business practice among cooperating enterprises contributing their resources (instruments, other facilities and operator personal) to create a Virtual laboratory that can run complex experiments on request from customers. To become consistent the DM should be

[3] http://www.collaboratory.nl/

supported by corresponding organisation of the access control infrastructure.

Figure 1 below illustrates relations between major hierarchical components in the DM resource management and security model. The following suggestions were used when creating this abstraction of the DM:

1) physically Instruments are located at the Facility but logically they are assigned to the VL and next allocation to the Experiment. Full context Instrument name will look like:

`CNL:Facility:VirtualLab:Experiment:InstrModel`

2) users/members of collaborative sessions are assigned to the Experiment, managerial and operator personnel belongs to VL and Facility and may have specific and limited functions in the Experiment;

3) particularly, domain based restrictions/policy can be applied to (dynamic) role assignment;

3) additionally, administrative rights/functions can be delegated by the superior entity/role in this hierarchical structure;

5) Trust Anchors (TA) can be assigned to hierarchical domain related entities to enable security associations and support secure communication. VL TA1 is suggested as minimum required in DM, Experiment TA2 may be included into the Experiment description. Collaborative session security association can be supported by AuthZ tickets.

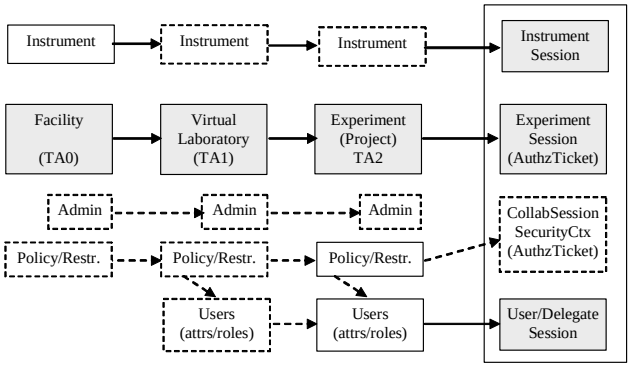

Figure 1. Domain based Resource management in SOCE

The Experiment description plays an important role in the DM security infrastructure, it is created by the experiment owner as a semantic object on the basis of a signed Experiment agreement (and in the context of the overall VL agreement). It contains all the information required to run the analysis, including the Experiment ID, assigned users and roles, and a trust/security anchor(s) in the form of the resource and, additionally, the customer's digital signature. The experiment description is used to provide experiment-dependent configuration data for other services to run the experiment and manage the dynamic security context. The Experiment description may also specify a workflow to orchestrate all interactions between experiment components/services and provide a solution for dynamic security context management as it is discussed in [5].

DM provides the following benefits:

1) reflects distributed hierarchical management model natural in distributed cooperative business environment;

2) multiple and hierarchical policies management that reflects hierarchical resource organisation;

3) allows for dynamic roles assignment with the domain defined restrictions;

4) supports dynamic security context management;

5) provides mechanisms for supporting multidomain authorisation sessions.

3. EXTENDING RBAC MODEL WITH DOMAIN BASED SECURITY CONTEXT MANAGEMENT

Fine-grained access control in typically interactive collaborative environment can be achieved using RBAC authorisation model, which generally consists of major functional components that include: Policy Enforcement Point (PEP), Policy Decision Point (PDP), Policy Authority Point (PAP) [6]. In RBAC, user/requestor access rights are defined by roles in a form of user attributes and a separately managed access control policy contains rules that define what roles are allowed to do what actions on the resource.

Generic RBAC model [6, 7] provides an industry recognised solution for effective user roles/privileges management and policy based access control. Many studies suggest RBAC as a natural method to model the security requirements in collaborative environment but at the same time they argue for application specific extensions, e.g., for user group organisation including additional group/team defined restrictions on separation of duties, roles/attributes combinations, etc. [8, 9].

Practical RBAC implementation requires resolution of many other administration and security related issues left out of scope in classical RBAC such as:

- policy expression and management,
- roles management and separation of duties,
- rights/privileges delegation,
- AuthZ session management mechanisms,
- security context management in dynamic scenario.

Two basic implementations of the generic RBAC model are Access Control Lists (ACL) that can be rather applications/implementation specific, and an emerging industry standard eXtensible Access Control Markup Language (XACML) that defines a rich policy expression format and simple Request/Response messages format for PEP-PDP communication [10]. XACML extensions and special profiles address most of mentioned above issues at the standard level. However, there are no widely used practical implementations for this new functionality.

The papers [11, 12] proposes an extension of the generic RBAC model the usage control (UCON) based

authorisation framework for collaborative application that specifically addresses access control to the consumable resources or which access should be coordinated among a group of users. This is achieved by using obligations, resource/environmental conditions, introducing mutable resource and user attributes, and applying ongoing control. The proposed implementation uses XACML as a policy expression language with proprietary defined the Obligation element. However, detailed analysis of the proposed UCON publications and implementations reveals that the UCON framework uses centralised policy management, environment and attributes control that may have a principal problem of races when using conditions/obligations on mutable attributes. Proposed usage session doesn't allow full functionality required for generic authorisation session management in a multi-domain environment.

When combining with the RBAC, the RBAC-DM (note, in most cases we will use abbreviations DM and RBAC-DM as equivalents) will intend to address most of above mentioned issues at the practical level by introducing domain related security context that actually reflects natural for cooperating entities/enterprises administration model and separation of duties. Use of Experiment and Collaborative session allows to implement delegations and minimum privileges principle in access control management but in its own turn requires consistent authorisation session context handling.

4. AUTHORISATION SERVICE OPERATION IN SOCE

In a SOCE, security services can be dynamically bound to main services at the service messaging level. This allows independent development but impose new requirements to the security services design:

- Orthogonal to basic services, e.g. achieved by providing generic security services and API's
- dynamically configurable both with structural components and environment context
- flexible security context management by separating security context from functional components and enabling dynamic services invocation
- capable of scaling over multiple security and administrative domains

A Resource or Service in SOCE is protected by site access control system that relies on both Authentication (AuthN) of the user and/or request message and Authorisation (AuthZ) that applies access control policies against the service request. It is essential in a service-oriented model that AuthN credentials are presented as a security context in the AuthZ request and that they can be evaluated by calling back to the AuthN service and/or Attribute Authority (AttrAuth). This also allows loose coupling of services (allowing domain independency even for hierarchical DM).

The Requestor requests a service by sending a service request ServReq to the Resource's PEP providing as much (or as little) information about the Subject/Requestor, Resource, Action as it decides necessary according to the implemented authorisation model and (should be known) service access control policies.

In a simple scenario, the PEP sends the decision request to the (designated) PDP and after receiving a positive PDP decision relays a service request to the Resource. The PDP identifies the applicable policy or policy set and retrieves them from the Policy Authority, collects the required context information and evaluates the request against the policy.

In order to optimise performance of the distributed access control infrastructure, the Authorisation service may also issue authorisation tickets (AuthzTicket) that confirm access rights. They are based on a positive decision from the Authorisation system and can be used to grant access to subsequent similar requests that match an AuthzTicket. To be consistent, AuthzTicket must preserve the full context of the authorisation decision, including the AuthN context/assertion and policy reference.

A typical DM access control use-case may require a combination of multiple policies and also multi-level access control enforcement, which may take place when combining newly-developed and legacy access control systems into one integrated access control solution. The SOCE experiments may apply different policies and require different user credentials depending on the stage of the experiment.

DM can improve overall services manageability but requires additional/corresponding mechanisms for dynamic security context management. It is also suggested that using AuthZ ticket with full session context will simplify distributed access control management in a hierarchical DM and allow for decoupling access control infrastructure components in a distributed environment.

Detailed analysis of how dynamic security context can be managed in SOA/Grid is discussed in the recent paper [5]. The following mechanisms and tools of the general access control infrastructure can be used to mediate a dynamic security context:

- Service and requestor/user ID/DN format that should allow for both using namespaces and context aware names semantics.
- Attribute format (either X.509/X.521 or URN/SAML2.0 presentation).
- Context aware XACML policy definition using the Environment element of the policy Target element (see section 5 for detailed discussion).

- Security tickets and tokens used for AuthZ session management and for provisioned resource/service identification. In both cases security tickets should contain the full security context and be supported by related AuthZ and provisioning infrastructure.
- Dynamic VO membership credentials (practically can be supported by existing VO management tools – see [13] for details) or other user and services federations.
- Workflow as primarily used for complex/combined services orchestration can be also used for managing dynamic security context.

Most of mentioned above mechanisms are available in complementary XML-based formats Security Assertion Markup Language (SAML) [14] and XACML [10] that are used for security assertion and access control policy expression.

5. ADDING WIDER SECURITY CONTEXT MANAGEMENT TO MAJOR AUTHZ FRAMEWORKS

To provide described above functionality on domain based security context handling, a number of features should be added to existing AuthZ frameworks such as Acegi Security [15], Globus Toolkit 4.0 AuthZ Framework (GT4-AuthZ) [16], gLite Java Authorisation Framework (gJAF) [17]. They are currently being developed as pluggable modules to a special GAAA-RBAC profile of the generic Authentication, Authorisation, Accounting (GAAA) Authorisation Framework (GAAA-AuthZ) [18, 19].

Acegi Security is industry recognised security solution with a particular emphasis on applications using Spring framework for J2EE (http://www.springframework.org/). It provides channel security, reach authentication and Single Sign-On (SSO) functionality, and also domain object authorization using Access Control List (ACL). Similar to GT4-AuthZ and gJAF, Acegi security services can be called from the main services using service and application specific filters.

GT4 Authorization Frameworks (GT4-AuthZ) is a component of the widely used Grid middleware that provides general and specific functionality to control access to Grid applications and resources using access control policies in Grid-specific formats, such as Access Control Lists (ACLs), gridmap file, identity or host based, and also providing external policy evaluation callouts using OGSA Authorization PortType that uses SAML as a messaging format. A simple XACML-based PDP is also provided.

gLite Java Authorisation Framework (gJAF) is a component of the gLite security middleware. It inherits compatibility with the early versions of the GT4-AuthZ that should ensure their future interoperability and common use of possible application specific modules. Both the GT4-AuthZ and gJAF services can be called from the SOAP based Grid services by configuring the interceptor module which operates in this case as a virtual PEP module together with the chain.

Similarity in interaction with the main services and applications provides a good basis for developing common modules/library to support dynamic and resource/application domain related context.

Figure 2 shows the GAAA-RBAC structure that contains the following functional components provided as a GAAAPI package to support all the necessary security context processing and communication between a PEP and a PDP:

- A Context Handler (CtxHandler) that calls to a namespace resolver (NS Resolver) and attribute resolver (AttrResolver), which in its own can call to external Attribute Authority Service (AAS) to validate presented attributes or obtain new ones.
- A Policy Information Point (PIP) that provides resolution and call-outs to related authoritative Policy Authority Points (PAP);
- Triage and Cache functionality that provides an initial evaluation of the request, including the validity of the provided credentials. This functionality is used for handling AuthZ tickets/tokens and also for AuthZ session management by evaluating service requests versus the provided AuthZ ticket/token claims;
- A Ticket Authority (TickAuth) generates and validates AuthZ tickets or tokens on the requests from PEP or PDP; to support AuthZ session tickets are cached directly by TickAuth or by PEP/PDP.

To support dynamic security context changes, the GAAAPI provides an advanced configuration management capability, based on the generic AuthZ service operational model. In particular, when the PEP function is invoked, during AuthZ request processing, it is dynamically configured with context aware modules NSResolver, Triage, TickAuth, and TrustDMngr.

When providing access control during a multi-stage experiment, the security context (e.g., the policies, team members and/or roles) may change. Such changes may be controlled in the experiment workflow and fed into access control system via an advanced configuration management interface to GAAAPI modules.

An AuthzTicket is generated as the result of a positive PDP decision. It contains the decision and all necessary information to identify the requested service. When presented to the PEP, its validity can be verified and in the case of a positive result, access will be granted without requesting a new PDP decision. Such a specific functionality is provided in the GAAAPI package with the Triage module.

The current GAAAPI implementation supports both SAML-based and proprietary XML-based AuthzTicket formats.

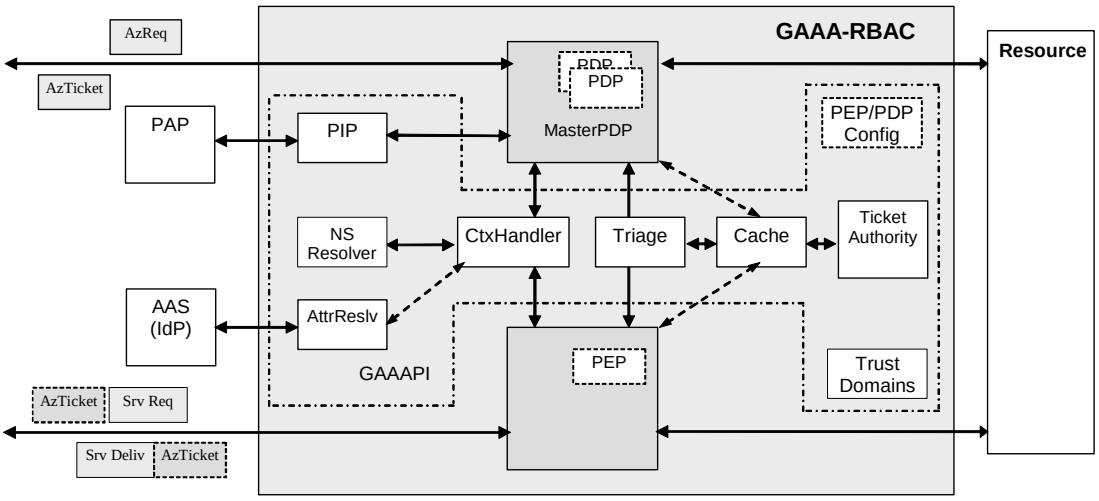

Figure 2. Security context handling in GAAA-RBAC and GAAAPI functional components

The AuthZ ticket and token handling functionality allows for performance optimisation and supports authorization session management. Further development includes extended AuthZ ticket format (both proprietary and SAML-based) to support multidomain provisioning scenarios and hierarchical resource and policy administration. Additional features include delegation and extended session context.

6. USING XACML FOR POLICY EXPRESSION IN RBAC-DM

A XACML policy is defined for the so-called target triad "Subject-Resource-Action" which can also be completed with the Environment element to add additional context to instant policy evaluation. The XACML policy format can also specify actions that must be taken on positive or negative PDP decisions in the form of an optional Obligation element. This functionality is important for potential integration of the access control system with logging or auditing facilities.

A decision request sent in a Request message provides context for the policy-based decision. The policy applicable to a particular decision request may be composed of a number of individual rules or policies. Few policies may be combined to form a single policy that is applicable to the request. XACML specifies a number of policy and rule combination algorithms. The Response message may contain multiple Result elements, which are related to individual Resources.

XACML policy format provides few mechanisms of adding and handling context during the policy selection and request evaluation. First of all, this is the policy identification that is done based on the Target comprising of the Resource, Action, Subject, and optionally Environment elements. Next, attributes identification and semantics can be namespace aware and used for attributes resolution during the request processing.

The DM makes extensive use of both XACML core specification and its special profiles for RBAC [20] and hierarchical resources [21]. Hierarchical policy management and dynamic rights delegation, that is considered as an important functionality in DM, can be solved with the XACML v3.0 administrative policy [22].

The XACML RBAC profile [20] provides extended functionality for managing user/subject roles and permissions by defining separate Permission `<PolicySet>`, Role `<PolicySet>`, Role Assignment `<Policy>`, and HasPrivilegeOfRole `<Policy>`. It also allows for using multiple Subject elements to add hierarchical group roles related context in handling RBAC requests and sessions, e.g., when some actions require superior subject/role approval to perform a specific action. In such a way, RBAC profile can significantly simplify rights delegation inside the group of collaborating entities/subjects which normally requires complex credentials management.

The XACML hierarchical resource profile [21] specifies how XACML can provide access control for a Resource that is organized as a hierarchy. Examples include file systems, data repositories, XML documents and organizational resources which example is the DM. The profile introduces new Resource attributes identifiers that may refer to the "resource-ancestor", "resource-parent", or "resource-ancestor-or-self".

XACMLv3.0 administrative policy profile [22] introduces extensions to the XACML v2.0 to support policy administration and delegation. This is achieved by introducing the PolicyIssuer element that should be supported by related administrative policy. Dynamic delegation permits some users to create policies of limited

duration to delegate certain capabilities to others. Both of these functionalities are quite relevant to the proposed DM and currently being investigated/tested.

Figure 3 below provides an example of the XACML policy which Target and IDRef bind the policy to the Resource. There may be different matching expression for the Resource/Attribute/AttributeValue when using XACML hierarchical resource profile what should allow to create a policy for the required resource hierarchy in DM. The example also contains the PolicyIssuer element that is related to the policy administration. In our example the the PolicyIssuer is declared as "`cnl:VLab031:trusted`" and in this case the PDP will rely on already assigned PAP and established trust relations. In case, when other entity is declared as a PolicyIssuer, the PDP should initiate checking administrative policy and delegation chain.

```
<PolicySet>
 <Target/>
 <Policy PolicyId="urn:oasis:names:tc:xacml:1.0:cnl:policy:CNL2-XPS1-test"
     RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:deny-overrides">
    <Description>Permit access for CNL3 users with specific roles</Description>
    <PolicyIssuer>
      <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
        DataType="http://www.w3.org/2001/XMLSchema#string">
        <AttributeValue> urn:oasis:names:tc:xacml:3.0:issuer:cnl:VLab031:trusted </AttributeValue>
      </Attribute>
    </PolicyIssuer>
  <Target>
    <Resources>
      <Resource>
        <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:anyURI-equal">
          <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#anyURI">
             http://resources.collaboratory.nl/Phillips_XPS1</AttributeValue>
          <ResourceAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
             DataType="http://www.w3.org/2001/XMLSchema#anyURI"/>
        </ResourceMatch>
      </Resource>
    </Resources>
  </Target>
<Rule RuleId="urn:oasis:names:tc:xacml:1.0:cnl:policy:CNL2-XPS1-test:rule:ViewExperiment" Effect="Permit">
    <Target>
      <Actions>
        <Action>
          <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
            <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
                 ViewExperiment</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">analyst</AttributeValue>
        <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">customer</AttributeValue>
        <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">guest</AttributeValue>
      </Apply>
      <SubjectAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:subject:role"
           DataType="http://www.w3.org/2001/XMLSchema#string" Issuer="CNL2AttributeIssuer"/>
    </Condition>
  </Rule>
 </Policy>
</PolicySet>
```

Figure 3. Example of XACML RBAC PolicySet containing PolicyIssuer element and hierarchical resource selector.

7. CONCLUSION AND SUMMARY

The results presented in this paper are the part of the ongoing research and development of the security infrastructure for user controlled multidomain services and its application to collaborative resource sharing. This work is being conducted by the System and Network Engineering (SNE) Group in cooperation with Telematika Institute in the framework of different EU and Dutch nationally and industry funded projects including EGEE, Collaboratory.nl, and GigaPort Research on Network. All of these projects intend to create and deploy interoperable security infrastructure at different levels.

The definition of the Domain based access control model RBAC-DM and proposed solutions described in this paper are based on the practical experience we have gained whilst designing and developing an open collaborative environment within the Collaboratory.nl project. RBAC-DM reflects distributed hierarchical management model typical for industrial collaborative

infrastructure and has additional features for domain related security context management. Use of Experiment and Collaborative sessions, supported by relevant session's security context management, allows for dynamic roles assignment with the domain defined restrictions, including delegation and minimum privileges principle.

The paper identifies major mechanisms that can be used for expressing and transferring dynamic security context in Grid and Web Services applications with extensive use of XML technologies. The implementation suggestions are given for how required context handling functionality can be added to popular AuthZ frameworks such as Acegi, GT4-AuthZ and gLite AuthZ frameworks. Proposed extension modules are being developed as a GAAAPI package of the GAAA-RBAC profile.

Proposed RBAC-DM and GAAAPI make extensive use of XACML core specification and its special profiles for RBAC and hierarchical resources, and also XACML v3.0 administrative policy. Example is provided of the policy using most of those features.

The authors believe that the proposed access control architecture for SOCE and related technical solutions will also be useful to the wider community has similar problems with managing access control to distributed hierarchically organised resources in dynamic/on-demand services provisioning.

8. REFERENCES

[1] Foster, I. et al, "The Open Grid Services Architecture, Version 1.0", Global Grid Forum, 29 January 2005, available from http://www.gridforum.org/documents/GFD.30.pdf

[2] "Web Services Architecture". W3C Working Draft 8 August 2003. - http://www.w3.org/TR/ws-arch/

[3] Demchenko, Y., L. Gommans, C. de Laat, A. Tokmakoff, R. van Buuren, "Policy Based Access Control in Dynamic Grid-based Collaborative Environment," The 2006 International Symposium on Collaborative Technologies and Systems, Las Vegas, May 14-18, 2006, Proceedings. IEEE Computer Society. ISBN: 0-9785699-0-3, pp. 64-73.

[4] Demchenko, Y., L. Gommans, C. de Laat, B. Oudenaarde, A. Tokmakoff, M. Snijders, "Job-centric Security model for Open Collaborative Environment," The 2005 International Symposium on Collaborative Technologies and Systems, Saint Louis, USA, May 15-19, 2005, Proceedings. IEEE Computer Society ISBN: 0-7695-2387-0, pp. 69-77.

[5] Demchenko, Y., L. Gommans, C. de Laat, A. Taal, A. Wan, O. Mulmo, "Using Workflow for Dynamic Security Context Management in Grid-based Applications," Grid2006 Conf. Barcelona, Sept. 28-30, 2006, Accepted.

[6] Information Technology - Role Based Access Control, Document Number: ANSI/INCITS 359-2004, InterNational Committee for Information Technology Standards, 3 February 2004, 56 p.

[7] Sandhu, R., Coyne, E. J., Feinstein, H. L. & Youman, C.E. 1996, "Role-Based Access Control Models", IEEE Computer, February 1996, pp. 38-47.

[8] Caelli W., Rhodes A., "Implementation of active role based access control in a collaborative environment", http://www.isi.qut.edu.au/research/publications/technical/qut-isrc-tr-1999-005.pdf

[9] Thomas, R. K. 1997, "Team-based Access Control (TMAC): A Primitive for Applying Role-based Access Controls in Collaborative Environments", Proceeding of the Second ACM Workshop on Role-Based Access Control, ACM, November 1997, pp. 13-19.

[10] Godik, S. et al, "eXtensible Access Control Markup Language (XACML) Version 2.0", OASIS Working Draft 04, 6 December 2004, available http://docs.oasis-open.org/xacml/access_control-xacml-2_0-core-spec-cd-04.pdf

[11] Park J.S., R Sandhu, "The UCONabc usage control model", ACM Transaction on Information and System Security, 7(1), February 2004.

[12] 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.

[13] Demchenko, Y., et al., "VO-based Dynamic Security Associations in Collaborative Grid Environment," The 2006 International Symposium on Collaborative Technologies and Systems, Las Vegas, May 14-18, 2006, IEEE Computer Society. IEEE Computer Society. ISBN: 0-7695-2387-0, pp. 38-47.

[14] Cantor, S. et al, "Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0," Committee Draft 04, 14 January 2005, available from http://www.oasis-open.org/committees/download.php/10627/sstc-saml-core-2.0-cd-03.pdf

[15] Acegi Security. - http://acegisecurity.org/

[16] GT 4.0: Security: Authorization Framework. [Online]. Available: http://www.globus.org/toolkit/docs/4.0/security/authzframe/

[17] Developer's guide for the gLite Java Authorisation Framework - https://edms.cern.ch/document/501718

[18] Vollbrecht, J., P. Calhoun, S. Farrell, L. Gommans, G. Gross, B. de Bruijn, C. de Laat, M. Holdrege, D. Spence, "AAA Authorization Framework," Informational RFC 2904, Internet Engineering Task Force, August 2000. ftp://ftp.isi.edu/in-notes/rfc2904.txt

[19] Generic Authorization Authentication and Accounting. [Online]. Available: http://www.science.uva.nl/research/air/projects/aaa/

[20] "Core and hierarchical role based access control (RBAC) profile of XACML v2.0", OASIS Standard, 1 February 2005, available from http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-rbac-profile1-spec-os.pdf

[21] "Hierarchical resource profile of XACML 2.0", OASIS Standard, 1 February 2005, available from http://docs.oasis-open.org/xacml/access_control-xacml-2.0-hier_profile-spec-cd-01.pdf

[22] "XACML 3.0 administrative policy," OASIS Draft, 10 December 2005. [Online]. Available from http://docs.oasis-open.org/access_control