# Security Architecture for Open Collaborative Environment

Yuri Demchenko[1], Leon Gommans[1], Cees de Laat[1], Bas Oudenaarde[1], Andrew Tokmakoff[2], Martin Snijders[2], Rene van Buuren[2]

[1] Universiteit van Amsterdam, Advanced Internet Research Group, Kruislaan 403, NL-1098 SJ Amsterdam, The Netherlands
{demch, lgommans, delaat, oudenaarde}@science.uva.nl
[2] Telematica Instituut, P.O. Box 589, 7500 AN Enschede, The Netherlands
{Andrew.Tokmakoff, Martin.Snijders, Rene.vanBuuren}@telin.nl

**Abstract.** The paper presents proposed Security Architecture for Open Collaborative Environment (OCE) being developed in the framework of the Collaboratory.nl (CNL) project with the intent to build a flexible, customer-driven security infrastructure for open collaborative applications. The architecture is based on extended use of emerging Web Services and Grid security technologies combined with concepts from the generic Authentication Authorization and Accounting (AAA) and Role-based Access Control (RBAC) frameworks. The paper describes another proposed solution the Job-centric security model that uses a Job description as a semantic document created on the basis of the signed order (or business agreement) to provide a job-specific context for invocation of the basic OCE security services. Typical OCE use case of policy based access control is discussed in details.

## 1. Introduction

The process industry makes extensive use of advanced laboratory equipment, such as electron microscopes, equipment for surface analysis and mass spectrometers. Laboratories tend not to have purchased highly specialized and sophisticated equipment, due to high initial outlay and operational costs and the expertise required to operate such equipment. The Collaboratory.nl[1] project (CNL) is one of the projects that investigate how technologies for remote operation of laboratory equipment can be integrated with existing Groupware and emerging Web Services and Grid technologies for enhanced remote collaboration.

This paper presents the results of the development of an open, flexible, customer-driven security infrastructure for open collaborative applications, in particular addressing practical needs of the Collaboratory.nl project. The proposed solution is based upon extended use of emerging Web Services and Computer Grid security technologies and the generic AAA authorisation framework [1, 2, 3, 4].

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

Collaborative applications require a sophisticated, multi-dimensional security infrastructure that manages the secure operation of user applications between multiple administrative and trust domains. Typical Open Collaborative Environment (OCE) use cases imply specifics of the collaborative environment that:

– is dynamic as the environment may change from experiment to experiment,
– may span multiple trust domains,
– needs to handle different user identities and attributes that must comply with different policies that are both experiment and task specific.

Managing access based upon role, assigned privileges and policy enforcement have been addressed in many collaborative and Computer Grids projects. It can provide a good basis for the development of security architecture for OCE. The majority of known solutions and implementations use widely recognised Role-based Access Control (RBAC) [5] model and its implementation in XACML [6] as a general conceptual approach.

The current Grid Security Infrastructure and Authorisation framework evolved from using proprietary systems like Community Authorisation Service (CAS) [7] to XACML based Policy Management and Authorization Service for Grid resources [8]. Although they provide a good example of addressing similar tasks, current Grid-based solutions cannot be directly used within OCE, since their deep embedding into parallel task scheduling mechanisms prevents distributed execution of dissimilar computational tasks/jobs. A typical collaborative environment is less coupled and mostly concerned with the allocation and execution of complex experiments on the equipment that for most use cases, requires human control and interaction during the experiment.

Collaborative tools such as CHEF[2], initially designed for online educational course management, can provide most of the necessary functionality for the creation of a collaborative environment. However, this environment needs to be extended such that it can be integrated with other stages and components of the collaborative organisation managing the experiment stages. These stages include the initial stage of order creation, and the main experimental stage that requires secure access to the instrument or resource.

To address the specifics, the proposed OCE Security Architecture uses a novel Job-centric approach, which is based on Job description as a semantic document, created on the basis of a signed order (business agreement). The document contains all the information required to run the analysis, including the Job ID, assigned users and roles, and a trust/security anchor(s) in a form of customer and/or OCE provider digital signature. In general, such approach allows binding security services and policies to a particular job or resource.

The paper is organized as follows. Section 2 of the paper describes basic OCE use cases and their required security functionality. Section 3 describes the general OCE Security Architecture and its general security services model. The architecture builds upon (and is intended to be compatible with) WS-Security and OGSA Security. Section 4 describes the OCE policy enforcement framework implementation using generic AAA architecture and RBAC based Authorisation service model.

[2] http://www.chefproject.org/

Proposed solutions are being developed in coordination with ongoing projects CNL and EGEE[3], and can represent a typical use case for the general Web Services and OGSA Security framework. It is expected that other similar projects such as VL-E[4] and GridLab[5] will benefit from this work as it intends to propose a general approach and common solutions for the security problems in OCE.

## 2. Basic OCE use cases and proposed Job-centric security model

Security services are defined as a component of the OCE middleware that provides secure infrastructure and environment for executing OCE tasks/jobs. Generally, security services can be added to an already existing operational architecture, however current industry demand for very secure operational environments requires that a Security architecture is developed as an integral part of the system design. There should be also a possibility to define a security services profile at the moment of a system service invocation defined by a security policy.

For the purpose of analysis of the required security functionality, all use cases in CNL can be split into two groups of simple security interactions and extended ones. In a simple/basic use case the major task is to securely provide remote access to instrument(s) belonging to a single provider. For this case, the remote site or the resource owner can provide few onsite services and allow distributed user groups. An extended use case must additionally allow distributed multi-site services, multiple user identities and attribute providers, and distributed job execution. In its own turn, multiple trust domains will require dynamic creation of user and resource federations/associations, handling different policies, specific measures for protecting data confidentiality and user/subject privacy in potentially uncontrolled environment.
In both cases, there is a need for the following functionality:
- fine grained access control based on user/subject attributes/roles and policies defined by a resource
- privilege/attribute management by a designated person carrying responsibility for a particular experiment or job
- customer controlled security environment with the root of trust defined by the customer (in particular, their private key).

Listed above functionalities require a new job-centric approach to security services provisioning in OCE that can be realised in the following way.

Procedures in OCE include two major stages in accepting and executing the order: negotiation and signing the order (business part), and performing the experiment (technical part). The Job description, as a semantic document, is created based on the signed order and contains all information required to run the experiment on the collaborative infrastructure. The job description contains the following components: a Job ID and other attributes, Job owner, assigned users and roles, business anchor(s)

[3] http://public.eu-egee.org/
[4] http://www.vl-e.nl/
[5] http://www.gridlab.org/

(BA) and/or trust/security anchor(s) (TA) in a form of customer and provider digital signatures.

Figure 1 illustrates a structure of the Job description and its relation to other OCE components and security services. This kind of job description can also be used as a foundation for creating Virtual Organisation (VO) [2] instance as an association of designated users and resources, which support all standard security constructs such as users, groups, roles, trust domains, designated services and authorities.

The job description (mandatory) must include or reference the Job policy, which defines all aspects of the user, resource and trust management when executing the job. This policy should define the following issues:

– trusted CA (Certification Authorities) or Identity Providers;
– trusted users, VO's, resources and in general trusted credentials;
– privileges/permissions assigned to roles;
– delegation policy;
– credit limits and conditions of use;
– confidentiality and privacy requirements;
– identity federation/mapping policy;
– Job access control or authorisation policy.

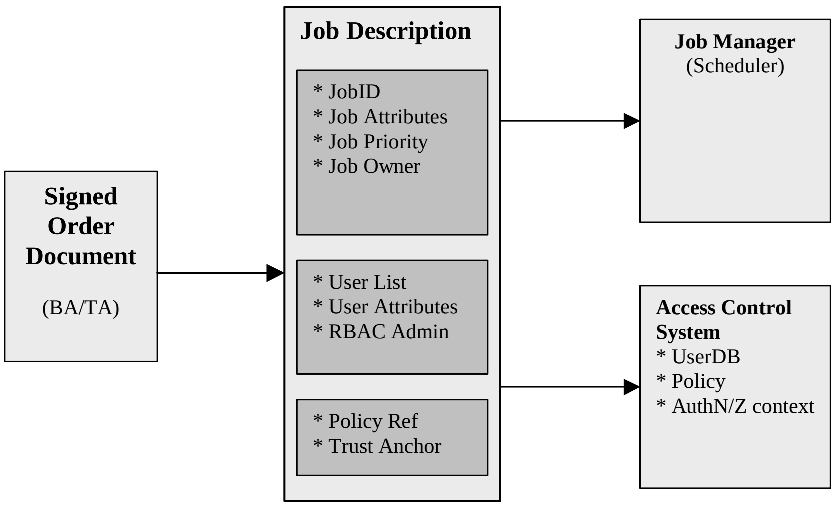

**Fig. 1.** OCE Security built around Job description

It is important to note that a Job policy may be combined with the Resource admission policy and in practice should not be more restrictive than the Resource policy. Otherwise the Job security management service may reject some resources based on Resource policy evaluation as a procedure of mutual authorisation.

Job-centric approach gives organizations complete flexibility in the creation of their security associations and services for the specific tasks or applications.

Practical implementation of the Job-centric security model requires wide spectrum of emerging XML and Web Services Security technologies that altogether constitute a general OCE Security Architecture as described in the next chapter.

## 3 Adopting Web Services Security Architecture for an Open Collaborative Environment

This section provides an overview and describes the general OCE Security Architecture based on wide use of the related WS-Security [1] and Open Grid Service Architecture (OGSA) [2] Security services and standards. The intension of this section is to provide guidance to existing XML and WS-Security standards and how they can be used for basic security services in OCE.

In Web Services Architecture (WSA) a Web Service is defined by PortTypes, Operations, Messages and Binding between all three components and actual back-end service in the form of WSDL (Web Services Description Language) description [9]. Security services and components can be added to the service description and defined by WS-Security set of standards. WS-Security components are already included into WSDP 1.4[6] and in the Globus Toolkit Version 3.2 and later[7].

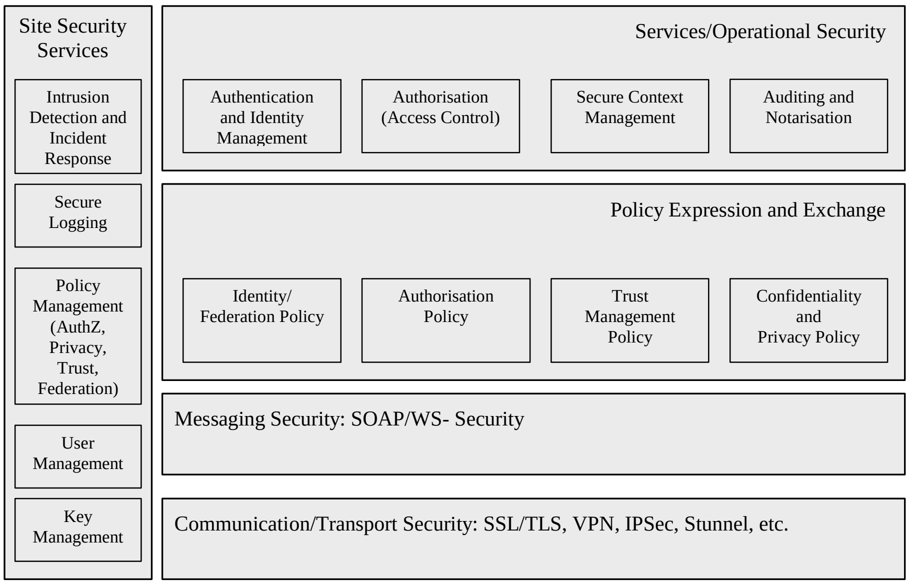
Site Security Services
Intrusion Detection and Incident Response
Secure Logging
Policy Management (AuthZ, Privacy, Trust, Federation)
User Management
Key Management
Services/Operational Security
Authentication and Identity Management
Authorisation (Access Control)
Secure Context Management
Auditing and Notarisation
Policy Expression and Exchange
Identity/ Federation Policy
Authorisation Policy
Trust Management Policy
Confidentiality and Privacy Policy
Messaging Security: SOAP/WS- Security
Communication/Transport Security: SSL/TLS, VPN, IPSec, Stunnel, etc.

**Fig. 2.** Security Architecture for an Open Collaborative Environment

The OCE Security Architecture is built upon above mentioned technologies and includes the following layers and components (see Fig. 2):

1) Communication/transport Security Layer defines network infrastructure security and uses such network security services as SSL/TLS, IPSec, VPN, and others.

2) Messaging Security Layer is based on currently well-defined SOAP/WS-Security mechanisms [10] and may use SAML as a security token exchange format [11].

[6] http://java.sun.com/webservices/jwsdp/
[7] http://www-unix.globus.org/toolkit/

3) Policy Expression and Exchange Layer defines set of policies which can be applied to OCE/VO users when they interact with the environment, and which are necessary to ensure multi-domain and multiplatform compatibility.

4) Services/Operational Layer defines security services/mechanisms for secure operation of the OCE components in an open environment: authentication and identity management, authorisation and access control, trust or secure context management, auditing/logging and notarization.

Some of the layers and components important for understanding the proposed OCE security architecture and job-centric security model are described in more details below.

## 3.1 Policy Expression and Exchange Layer

Communicating OCE services/principals need to conform to certain requirements in order to interact securely. It is important that service or resource requestors have access to and understand policies associated with the target service. As a result, both the service requestor and service provider must select an acceptable security profile. It is also important to mention that the privilege to acquire security policy is given by the hosting environment to authenticated and authorised entities only.

The policy layer provides necessary information for the policy enforcement modules of the Operational Security layer. It is suggested that policy expression should conform to the general WS-Policy framework and may include component policies using different policy expression formats including XACML [6] or Generic AAA [12] policy formats. Policies for end applications and services are described by (a set of) rules for a specific target comprising of a triad (Subject, Resource, Action). Policy may also include common attributes such as security tokens, delegation and trust chain requirements, policy combination algorithms in a form defined by WS-Security and WS-Policy [13].

Policy association with the service can be done using WS-PolicyAttachment that defines extensions mechanisms to attach a policy or policy reference to different components of the service description including PortType, operation/message, or arbitrary element [14]. XACML Web Services profile defines the mapping between Web Service description components and its own policy description hierarchy [15].

## 3.2 Authentication and Identity Management

The OCE operational environment may consist of multiple locations and security domains that maintain their own Authentication and Identity management services. Interoperation and secure Single Sign-on (SSO) will require federation of the involved domains and identity and credentials translation or mapping that can be built on two currently available identity management specifications: WS-Federation [16] and Liberty Alliance Project (LAP) [17]. Preferences can be defined by other OCE security components. LAP has benefit of being a more independent implementation, but it implements more business oriented approach. WS-Federation is more naturally integrated with Web services environments.

Authentication and Identity management services can use PKI based credentials for user/entity identification and authentication, and SAML and WS-Security for security credentials and token definition and exchange format [10, 11].

### 3.3 Authorisation and Access Control

The Authorisation and Access Control security service is a key part of the managed security in an open service oriented environment. Authorisation is typically associated with a service provider or resource owner. The resource owner controls access to a resource based upon requestor credentials or attributes that define the requestor's privileges or associated roles bound to the requestor's identity. Separation of Authentication and Authorisation services allows dynamic RBAC management [5] and virtual association between interacting entities, and provides a basis for privacy.

For distributed multi-domain services/applications, an Authorisation service can operate in pull or push modes as defined by the generic AAA architecture [3, 4] in which correspondently Authorisation decision is requested by a Resource service, or requestor preliminary obtains the Authorisation token from the trusted Authorisation service. It subsequently presents the token together with the authorisation context to the resource or service. When using the push or the combined pull-push model for complex services requests, the SAML format is considered as a recommended format for authorisation tokens exchange.

## 4 Policy Based Access Control using RBAC model and Generic AAA framework

This section provides an illustration how generic security components, in particular generic AAA framework and RBAC, can be used for providing policy based access control functionality in OCE.

Security services may be bound to and requested from any basic OCE service using a standard request/response format. Security services use must be specified by the policy that provides a mapping between a request context (e.g., action requested by a particular subject on a particular resource) and resource permissions.

Binding between (core) services and security services can be defined dynamically at the moment of service deployment or invocation using existing Web services and XML Security technologies for binding/associating security services and policies to the service description as explained above.

Figure 3 illustrates basic RBAC components and their interaction during the evaluation of the authorisation request against the access policy. When combined with the Job- centric approach, the Policy components and attributes and the request context can be defined by the particular job description that binds job attributes, user information and established trust relations between the customer and the provider.

The OCE policy/role based Authorisation infrastructure consists of

– a PDP (Policy Decision Point) as a central policy based decision making point,

– a PEP (Policy Enforcement Point) providing Resource specific authorisation request/response handling and policy defined obligations execution,
– a PAP (Policy Authority Point) as a policy storage (in general, distributed),
– a PIP (Policy Information Point) providing external policy context and attributes to the PDP including subject credentials and attributes verification
– a RIP (Resource Information Point) that provides resource context
– a AA (Attribute Authority) that manages user attributes and can be in particular case VO management service (VOMS) [19].

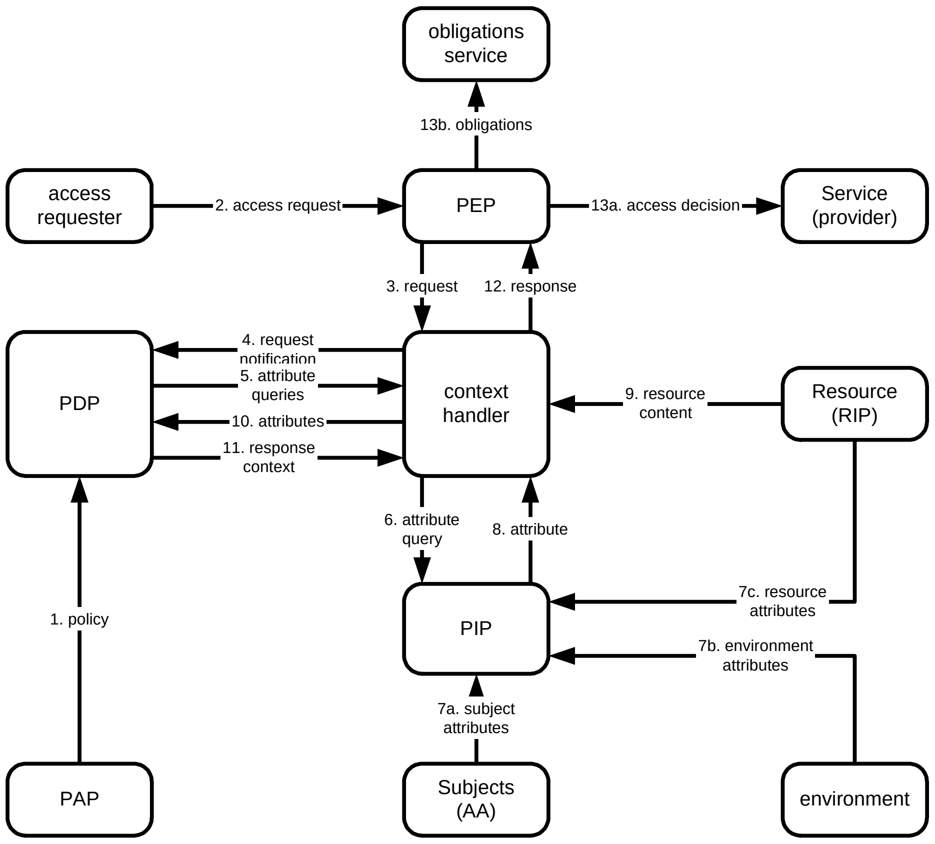

**Fig. 3.** RBAC based authorisation system components and dataflows.

To obtain a permission to access a resource, a user or resource agent request via PEP an authorisation decision from PDP that evaluates the authorisation request against the policy defined for a particular job, resource and user attributes/roles. The access policy is normally defined by the resource owner and may be combined with the Job policy. The PEP and PDP may also request specific user attributes or credentials from the authentication service, or additional information from the Resource.

When using XACML as a policy and messaging format, the Policy is constructed as a set of rules against the Target defined as a triad (Subject, Resource, Action), and the Request message also requests a PDP decision against a particular Target. The Subject element may contain Subject ID, Subject authentication or identity token, attributes or role(s), and other credentials. The Response message returns the PDP decision and may contain obligation conditions that must be executed by the PEP.

## 6 Summary and Conclusions

The general OCE Security architecture and implementation suggestions described in this paper are based on practical experience of building open collaborative environment for the Collaboratory.nl project undertaken in framework of the Collaboratory.nl consortium.

Proposed Security Architecture is based upon existing and emerging Web Services Security technologies and intends to be compatible with the OGSA Security Architecture. Existing XML and Web Services Security technologies provide a basis for building distributed Security infrastructure for OCE and easy integration with existing and developing security services and solutions. Use of industry standards such as WS-Security, XACML and SAML will guarantee future compatibility between CNL/OCE implementations and third party products for security services.

On other side, the CNL project provides a good platform for testing and further development of the proposed ideas and solutions. The CNL Security Architecture implements proposed Job-centric approach that allows building basic CNL security services around the semantic Job description document created on the base of signed order and containing all information required to run the experiment or execute the job on the CNL and enable basic security services such as user authentication, policy and role based access control, confidentiality and integrity of information and data.

The CNL Authorisation framework combines Web Services security mechanisms with the flexibility of the Generic AAA Architecture and XACML policy/role based access control model to built fine-grained access control. Separating policy definition from the authorisation or access control execution/enforcement will simplify access control management, which can be delegated to the resource owner. To reduce performance overhead when requesting authorisation decision from PDP, CNL implementation combines pull and push models [4] by using authorisation ticket with the limited validity period that allows to bypass potentially slow request evaluation by PDP.

The CNL project is being developed in coordination with the EGEE project what will allow future use of the Grid infrastructure being development in the framework of EGEE project and guarantee the compatibility of basic security services such as authentication, authorisation, and corresponding formats of metadata, policies, messages, etc.

Authors believe that the proposed OCE Security architecture and related technical solutions will be interesting for other projects dealing with the development of middleware for virtual laboratories and collaborative applications. Discussed here OCE specific components will be available openly as a part of the GAAA Toolkits by the middle of 2005.

## Acknowledgements

This paper results from the Collaboratory.nl project, a research initiative that explores the possibilities of remote control and use of advanced lab facilities in a distributed and collaborative industrial research setting. The Collaboratory.nl consortium consists of DSM, Philips, Corus, FEI, Telematica Instituut and the University of Amsterdam.

## References

1. Security in a Web Services World: A Proposed Architecture and Roadmap, Version 1.0, A joint security whitepaper from IBM Corporation and Microsoft Corporation. April 7, 2002, http://www-106.ibm.com/developerworks/library/ws-secmap/
2. The Open Grid Services Architecture, Version 1.0 – 12 July 2004, 2004. - http://www.gridforum.org/Meetings/GGF12/Documents/draft-ggf-ogsa-specv1.pdf
3. RFC 2903 , Experimental, "Generic AAA Architecture",. de Laat, G. Gross, L. Gommans, J. Vollbrecht, D. Spence, August 2000 - ftp://ftp.isi.edu/in-notes/rfc2903.txt
4. RFC 2904 , Informational, "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
5. Role Based Access Control (RBAC) – NIST, April 2003. - http://csrc.nist.gov/rbac/
6. eXtensible Access Control Markup Language (XACML) Version 1.0 - OASIS Standard, Feb. 2003 - http://www.oasis-open.org/committees/download.php/2406/oasis-xacml-1.0.pdf
7. K.Keahey, V.Welch, “Fine-Grain Authorization for Resource Management in the Grid Environment”. - http://www.fusiongrid.org/research/papers/grid2002.pdf
8. Markus Lorch, Dennis Kafura, Sumit Shah, "An XACML-based Policy Management and Authorization Service for Globus Resources". - Grid 2003, 17 November 2003. - http://zuni.cs.vt.edu/publications/grid-authz-policy-mgmt-wip03.ps
9. Web Services Architecture, W3C Working Draft 8 August 2003 - http://www.w3.org/TR/ws-arch/
10. Web Services Security Framework by OASIS - http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss
11. Security Assertion Markup Language (SAML) v1.0 - OASIS Standard, Nov. 2002 - http://www.oasis-open.org/committees/documents.php?wg_abbrev=security
12. A grammar for Policies in a Generic AAA Environment - http://www.ietf.org/internet-drafts/draft-irtf-aaaarch-generic-policy-03.txt
13. Web Services Policy Framework (WS-Policy). Version 1.1. - http://msdn.microsoft.com/ws/2002/12/Policy/
14. Web Services Policy Attachment (WS-PolicyAttachment). Version 1.1. - http://msdn.microsoft.com/ws/2002/12/PolicyAttachment/
15. XACML profile for Web-services (WSPL): - http://www.oasis-open.org/committees/download.php/3661/draft-xacml-wspl-04.pdf
16. Web Services Federation Language (WS-Federation) Version 1.0 - July 8 2003 – http://msdn.microsoft.com/ws/2003/07/ws-federation/
17. Liberty Alliance Phase 2 Final Specifications - http://www.projectliberty.org/specs/
18. Demchenko Yu. Virtual Organisations in Computer Grids and Identity Management. – Elsevier Information Security Technical Report - Volume 9, Issue 1, January-March 2004, Pages 59-76.