

# Unlocking Repositories: Federated Security Solution for Attribute and Policy Based Access to Repositories via Web Services

Marek Hatala, Ty Mey (Timmy) Eap and Ashok Shah  
*School of Interactive Arts and Technology*  
*Simon Fraser University*  
*Surrey, BC, Canada, V3T 2W1*  
*{mhatala, teap, ashaha}@sfu.ca*

## Abstract

*In this paper we describe our novel solution for web services enabling users from the trusted organizations to access learning objects in the repository based on their attributes in their home organizations. The solution extends a web-based Shibboleth system into the realm of web services. It utilizes the Web Services Security SAML profile and combines it with the XACML access control policies. The technical solution is described in the context of the Course Management Systems with complex access policies in operation at our campus.*

**Keywords:** federated security, trust, attribute-based access control, web services, repositories

## 1. Introduction

Effective sharing and interoperability have become crucial for conducting business and providing services in many sectors such as government research, education, and business in general. Although many resources can be shared freely there are many cases where the access to the resources has to be limited to a certain group of individuals. A scalable solution for secure access to resources is becoming one of the main enablers for greater interoperability in the web environment.

In this paper we present our results in building a secure infrastructure for the network of object repositories connected via Web Services. The infrastructure adds support for security through the placement of a security layer on top of the digital repository interoperability layer. We have defined and implemented several security profiles for controlling access to the repositories either for individual users or group of users from trusted institutions characterized by their attributes.

Our design is compatible with the current security solution for Web-based applications and extending it to rich clients and Web Services. The implementation uses the latest standards developed by the Web Services Security group at OASIS and Internet2 Shibboleth working group.

The remainder of the paper is organized as follows. In Section 2, we provide a motivation for our work. In Section 3 we provide background of new technologies we build on and in Section 4 we the idea of trust federations. In Section 5 we describe our solution to federated access. Section 6 provides an account on implementation and deployment. We compare our work with other approaches in the field in Section 7 and conclude in Section 8.

## 2. Motivation

The Course Management System (CMS) at Simon Fraser University (SFU) Surrey is used to manage and deliver courses that are structured into learning objects<sup>1</sup>. Typically, one week activity contains a web presentation which in turn consists from 3-5 segments each represented by a rich web content, assignment specification, in-class activity description and guidelines, and optionally links to the discussion forum and other specialized objects such as co-constructive segments, etc. The presentation segments are true learning objects that were developed by faculty and directly include or link to other documents. These can be cached copies of materials from the Web for which copyrights have been cleared with the material's owners; referenced materials from the digital library maintained by the SFU library with different licensing

---

<sup>1</sup> IEEE Learning Object Metadata Specification defines learning object as "any entity, digital or non-digital, which can be used, re-used or referenced during technology supported learning" (<http://ltsc.ieee.org>).

agreements with publishers; and material directly purchased from publishers for specific course offerings.

From the access control perspective majority of the faculty is comfortable sharing their own learning objects with general public. However, to complicate the matters, learning objects, quite often, include resources that each differ in access rights, causing the learning object to have combined access rights. For example, a unit of instruction prepared by a professor who wants to share it with the community can contain an image from an image library with a license allowing access to any member of the academic community at a Canadian university and a scanned image from a book with access purchased from a publisher for a specific group of students taking a course. The level of access for each object is carefully maintained by a licensing specialist for the purpose of managing licenses and renewing them as necessary.

To restrict the access to the objects with the most restrictive access policies (about 10 percent of the repository content) the current solution conservatively locks the whole repository and only students and instructors directly associated with the courses can access the material after they log in to the system. This makes the remainder of the repository content inaccessible to larger academic community and general public.

One solution enabling SFU community access to resources within the CMS could be accomplished by binding the access policies to the directory information available for SFU users. In this way the access could be extended for example to the library resources where SFU faculty members have an access based on the library's licensing agreements. However, this solution would not help the rest of the academic community even if the resources are free or the users may have an appropriate license.

An ideal solution would reflect the granularity of the access policies as specified for the individual learning objects within the repository. In CMS, the object access policies reflect the roles people embody when they communicate with the repository, such as faculty-at-sfu, faculty-at-canadian-university, general-public, undergraduate-registered-to-itec426-fall2005, etc.

Our solution is based on two concepts: user attributes and federated security. The solution turns the CMS in to the repository providing the users with the broadest access to the resources based on the *attributes* the users obtain from their home institutions. The attributes can represent user roles, such as 'role=graduate student', organization attributes such as 'organization=sfu', or more user specific information

such as 'email=mhatala@sfu.ca'. The solution matches the values of the attributes against the access control policies. The concept of *federation* provides the system with the infrastructure information which organizations issuing the attributes to trust.

In the following sections we describe each aspect of the solution in detail.

### 3. Tackling Complexity with Attributes

An ideal solution would provide each individual user with the appropriate level of access. However, even a moderate number of users creates difficulties in maintainability [9, p. 113]. Role-based access control [7] is widely used as an alternative in operating systems, databases and web services. In this approach access rights are assigned to the roles. Roles could be organized in hierarchies, creating a manageable access rights system. Individual users are assign the roles with associated access rights depending on the roles they play and tasks they perform in the organization.

Although the role-based access is a good solution for the systems where individual identity of the users is known to the system upfront it does not scale in the distributed context.

An alternative is to grant access to resources based on the user characteristics such as employer, status within the organization, project assignment, etc. These characteristics are called *attributes* and to be useful they have to be generally recognized within the larger community providing access to its distributed resources. This approach is particularly attractive in the context of loosely coupled distributed systems such as web and web services. Several technologies and standards have been developed that utilize the notion of the attributes for enabling access control decisions. In this section we briefly review those that are relevant to our research: XACML and SAML.

#### 3.1 Access Policies and XACML

To protect the resource on the system we have to specify the rules who has the right to access the resource and what operations the user can perform on the resource. This set of rules is called a policy. Extensible Access Control Markup Language (XACML) XACML defines a general policy language used to protect resources as well as an access decision language [8].

Specifically, XACML allows creating policy rules with conditions as a logical expression combining attributes of the subject and/or resource. For example, a policy can specify that only the users with the attribute 'organization' having values from the 'edu' domain can access a particular resource. The appropriate level of access for each user is determined

by the policies instead of the direct specification on per-resource basis.

One of the benefits of the XACML is that it allows developers to delegate policy execution to generic policy engines, such as Sun's XACML Implementation<sup>2</sup>.

### 3.2 Attributes and SAML

While XACML provides means for making an access decision (possibly based on user attributes) it does not provide a way how to deliver those attributes between loosely coupled systems. A standard called Security Assertion Markup Language (SAML)<sup>3</sup> allows entities to make assertions regarding the identity and attributes, and entitlements of a user to other entities, such as a partner company or web service. SAML assertions can be used within SOAP messages in order to carry security and identity information between actors in web service transactions. The SAML Token Profile for the Web Services Security<sup>4</sup> specifies how SAML assertions should be used with web services. At the time of our project there was no stable reference implementation of this profile.

## 4. Building Trust with Federations

Carmody et al [6] define the *federation* as “an association of organizations that use a common set of attributes, practices and policies to exchange information about their users and resources in order to enable collaborations and transactions”. The federation is basically a framework supporting sharing between organizations who agree to endorse federation's policies and implement technical agreements.

The Liberty Project<sup>5</sup> is a consortium of over 150 organizations that “is working to address the technical, business, and policy challenges surrounding identity and Web Services or federated *identity management*”. The project is currently in development of specifications, guidelines and best practices. The Liberty identity federation framework proposes the use of a federated network identity to solve the problems of identity management [3]. The Shibboleth Project has developed an open source system that provides a federated security for Web-based applications. In Shibboleth, providers make an authorization decision based on the attributes issued to the users by their *home organization*. The attributes and their values are agreed upon by the federation members and are

exchanged in the form of SAML assertions (see Section 3.2).

### 4.1 InCommon Federation and Shibboleth

“The mission of the InCommon Federation is to create and support a common framework for trustworthy shared management of access to on-line resources in support of education and research in the United States.”<sup>6</sup> InCommon is built using Shibboleth authentication and authorization technology for access to web resources.

Shibboleth<sup>7</sup> is an initiative by Internet2 working to develop security solutions that promote inter-institutional collaboration and access to digital content. It is a Single-Sign-On (SSO) and attribute-based exchange protocol consisting of two main components: Identity Provider (IdP) and Service Provider (SP). For the installation of the Shibboleth IdP, it is recommended to have a secure local authentication system and an LDAP directory, storing user attributes in eduPerson organizational scheme. The IdP is coupled with Attribute Authority (AA), which maintains a set of policies called Attribute Release Policies that govern the sharing of user attributes with Shibboleth SP sites [2].

In a SSO model, users sign onto their respective IdPs and then they can access all SPs in the federation. Shibboleth achieves SSO by using Web browser technologies: redirection and cookies. When a user accesses services on a SP, for the first time, the SP displays a list of trusted IdPs<sup>8</sup>. The SP then redirects the user to the selected IdP with appropriate parameters necessary for the IdP to validate the SP as a trusted entity and redirect the user back to the SP once the authentication is complete. The information is passed between the IdP and SP in the cookies, and the user's profile is delivered directly to the SP from the IdP in the form of SAML attribute assertions [4].

### 4.2 Web Services and Federated Security

Web Services differ from browser transactions in that they lack the user interactions in the addition to redirection and cookies. User applications access Web Services on behalf of the users. The applications can determine the user profile's attributes required to access a service before making the request. Hence, the applications can obtain these attributes from an IdP

<sup>2</sup> <http://sunxacml.sourceforge.net/>

<sup>3</sup> <http://www.oasis-open.org/committees/security/>

<sup>4</sup> Accessible from <http://www.oasis-open.org/committees/wss>

<sup>5</sup> <http://www.projectliberty.org>

<sup>6</sup> <http://www.incommonfederation.org/about.cfm>

<sup>7</sup> <http://shibboleth.internet2.edu>

<sup>8</sup> Trust is maintained by the federation. For example, in the InCommon federation sponsored by Internet2 to service US higher educational institutions, all SPs and IdPs receive a list of all trusted sites and certificates to validate trust and authorize the access control to digital contents.

and use it to access the service on a SP. To make the operation safe, a combination of security features such as certificates, signatures and encryption has to be utilized and treated in a specific way that is defined in the security profile. Our work defines the necessary infrastructure components and profile to support the Web Service security in the federation of Web Services.

## 5. Federated Security Solution

ECL is a Web Services based middleware infrastructure that connects learning object repositories and services throughout Canada, USA, UK, Australia, and Europe [5]. The main characteristics of the ECL infrastructure include:

- Easy connection of new services to the network via provided middleware
- End user discovery and instant use of newly added services
- Support for federating queries to both ECL services and services outside of ECL network.

The ECL consists of ECL Protocol, ECL Connector (a connecting middleware), ECL Service Registry for discovery of services, ECL Gateway (a middleware framework for building bridges between ECL network and non-ECL services). The ECL defines four types of requests identified by IMS DRI functions (search, gather, submit, and request) and their corresponding responses. The ECL Protocol is implemented in the form of document style web services where requests are defined in the form of messages with their corresponding XML schemas. The ECL messages are delivered in the body of the standard SOAP messages.

The ECL security infrastructure added two new infrastructure components: ECL Certification Authority and ECL Identity Provider. It also extended the role of the ECL Service Registry. In fact, the solution is independent of the ECL protocol as it utilizes the SOAP messages header and leaves the message content unchanged<sup>9</sup>.

### 5.1 Certification Authority

The ECL-SI solution depends on certificates to be issued to users and repositories by trusted entities. Unlike Web application, all members including users must have a certificate signed by a trusted CA. The ECL-SI accommodates both a commercial CA and its own ECL CA. The ECL CA can be integrated into the

organizational authentication system and will provide certificates to the organization members and services.

The ECL-SI supports the establishment of trust in a small group of organizations by organizing their CAs into a hierarchy and providing mechanism for validating the trust for all members under the same CA umbrella. It should be noted, that the strength of the security depends on the authentication system, and it is up to the federation to decide what authentication system is appropriate for their federation.

The certificates issued to ECL users are short-lived certificates. Two types of certificates are used: identity certificate and opaque certificate. The identity certificate is used to communicate with the ECL AA (see below). The opaque certificate is used when ECL client opens mutually authenticated SSL connection to other ECL services.

ECL CA has been implemented as a plugin to Shibboleth IdP and utilizes its codebase. More importantly it also utilizes the same trust fabric of the federation the Shibboleth deployment is part of.

### 5.2 Identity Provider

Identity Provider is an organizational entity that authenticates users and issues attribute assertions to the users or to service providers that users want to access. The assertions guarantee that the users are members of the organization and are digitally verifiable. The ECL IdP was developed as a plug-in to Shibboleth IdP. It extends current Shibboleth IdP to enable the support for web services. ECL IdP includes three components: a certificate authority (CA), an attribute authority (AA), and an impersonation service (IS). These three plug-in components provide a secured mechanism for the delivery of the SAML assertions to service providers including the delivery over a proxy. In ECL-SI, client applications authenticate users and obtain certificates from CA at startup. When the users want to access a service provider, their application determines the required attributes, formulates a SAML attribute query request, and uses the identity certificate to make the call to the AA. The AA is bound to the organizational directory service. So it has access to the user information.

ECL AA is slightly different from Shibboleth AA. It must authenticate the certificate used by the client application for opening SSL connection before releasing the SAML attribute assertion. In addition, it inserts the user's opaque certificate as the holder of the assertion. The receiver of the assertion compares this embedded certificate with the certificate the client application used for SSL connection. This technique is

---

<sup>9</sup> Unless encryption is requested when the message content is replaced with its encrypted form.

called holder-of-the-key method<sup>10</sup>. It ensures that the assertion belongs to the person who makes the request. Impersonation Service is used when the request is made over a proxy or a gateway. To allow a gateway relay the request to a service provider, the gateway must ask IS to include its certificate as holder of the assertion.

The ECL IdP differs from the standard Shibboleth IdP in several ways. Shibboleth IdP is contacted by the server providers who have to be trusted and releases the user attributes based on the attribute release policy specific for each service provider. ECL IdP communicates directly with the user who request attributes from the IdP. It is up to the user to decide which attributes s/he agrees to release to the service provider.

### 5.3 Service Registry

A service registry (SR) is a component that makes the infrastructure highly flexible and intelligent. The ECL SR holds information about service providers, supported security profiles, their system certificates, and required attributes to access resources. For example, the record in the registry can specify that the particular repository supports federated security profile (see below), trusts IdPs at SFU and PenState, and requires eduPerson Principal Name and eduPerson Scoped Affiliation attributes. Based on this information, the user's application can determine if the user has required security credentials to access resources on a particular service provider.

To access a resource, the user's application begins by determining if the service provider is a trusted member of the federation, and then checks if its IdP is part of the same federation or is in the list of the IdPs specified for this service. Finally, it checks if it has all the required attributes and values requested by the SP. These checks are done automatically and can be used to pre-select a set of services that the application is able to communicate with. The user makes the final selection from the list of compatible services and approves a release of the attributes for each service.

In the ECL scheme, users do not have to know the service providers upfront. They can discover the services in the registry. This is different from other federations such as InCommon [6] where the federation maintains the list of trusted entities and the list has to be distributed to all members of the federation.

---

<sup>10</sup> The method opens the SSL connection using trusted certificate only with the holder of the matching private key. As a result the connection is established only with the agent with the proven identity.

### 5.4 Federated security profile

A security profile is a description of a set of security tokens defined by the ECL-SI. It is a messaging protocol between client applications and the SPs. ECL-SI defines two security profiles for the SOAP-based protocol. The profiles are defined based on the WSS standard [1]. The message security level is achieved through the combination of security tokens in the SOAP header, encryption, and digital signatures of the message. A combination of different security techniques within the profile allows the ECL-SI to achieve different levels of security. This is different from the approach used by SSL where the entire communication is over a secured wire. As a result, the messages in the ECL-SI can be safely delivered over the unsecured wire and can be (partially) processed by the middleware services if required. ECL Security defines two profiles:

- Repository controlled security profile
- Federated security profile

In this paper we do not deal with the repository controlled profile. The technical details for both profiles can be found at the ECL website (<http://ecl.iat.sfu.ca>).

The federated security profile supports security and privacy of communication between two parties that trust each other, based on their membership in the trust federation. This means that the access to their organizational resources is based on the security attributes issued by another organization.

Figure 1 shows the communication flow of messages and information passed between a client, a repository and infrastructure services. The client discovers the services via the registry. The client authenticates and requests his security attributes from his home institution. The repository validates the attributes and makes an authorization decision based on the user's attributes and the access control policies. As a result, critical user information is kept at their home organization.

In addition the security tokens in the profile are encrypted and signed by the issuers to avoid possible attacks. Furthermore, timestamp is mandatory on SAML assertions. The signature by itself avoids the hijacking of the security parameters before the ECL message is formed. The timestamp makes it impossible to reuse tokens after it expired.

To avoid the man-in-the-middle attack, a combination of the timestamp, signature and encryption is used between the client application and SP repository. The encryption ensures that the security parameters are not readable to someone who hijacks

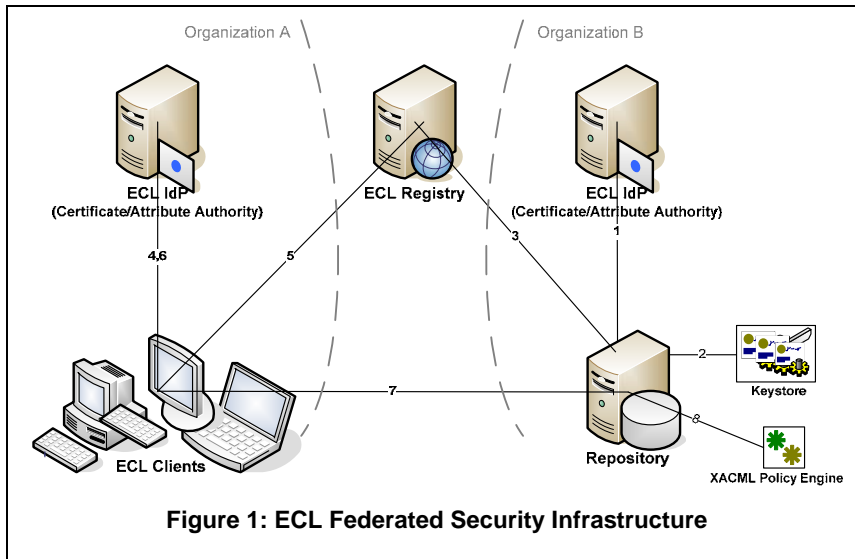


Figure 1: ECL Federated Security Infrastructure

the SOAP message. Before the hijackers can extract and decrypt security tokens from the SOAP message, the security tokens would be expired and become useless to the hijackers.

### 5.5 XACML based Policy Engine

Although the access control decision falls outside of the scope of ECL security solution and depend on the specific repository situation we have developed the XACML-based policy engine that can be adopted when a new scalable and maintainable solution is needed. We use a standard implementation of XACML policy engine from Sun and connect it to the federated security infrastructure as described above. Three specific extensions were needed.

First, we generate access policies based on the information that is stored and managed in the relational database. This enabled us to leverage the access control structures and processes that were in place in the existing system. The transformation itself is rather straightforward. However, when the policies are generated and how are they managed by the policy engine affects scalability significantly. One option, which was rejected after initial testing, was to generate all policies at the start up time and load them into the policy engine. The solution we adopted was to pre-generate all the policies but to keep them in the database. Only the policies for the resources affected by the incoming request are loaded into the policy engine and executed.

Second aspect was interconnection between SAML attribute language used to communicate user's

attributes and the XACML query language. This was resolved by developing the specific XSLT converter<sup>11</sup>.

The third, and the most interesting problem, dealt with and issue of implicit attribute values. For example, when user request contains attributes that the user is employee of PenState (i.e. has attribute 'psu') and XACML policy states that a particular resource can be access by members of the employees from the 'edu' we have to extend original attributes with missing values. We have developed a rule engine that extends provided

user attributes with implicit values as agreed upon within the federation.

### 6. Implementation and Deployment

The ECL-SI is a middleware component that encapsulates and abstracts Web Service implementation from developers. The implementation utilizes mainstream Java Web Service implementations such Axis SOAP engine, Shibboleth, OpenSAML<sup>12</sup>, and WSS4J<sup>13</sup>.

To instantiate federated security, the repository developers configure ECL-SI connector to accept users from a federation and/or from an IdP by adding trusted certificate of the IdP and/or the root CA certificate of the federation into the list of trusted IdPs. In addition, developers can specify what attributes are required to access their repository.

Any type of certificate can be used for the repository. However, the certificate or the root CA certificate of the repository must be added into the list of the clients' trusted service providers. It is important to note that when adding a root CA certificate into the list, the clients will trust all the repositories that have certificates rooted from that root CA certificate. This trust verification technique is powerful, but the federation must have a dedicated root CA. A trusted federation can negotiate a deal with a commercial CA such as VeriSign to issued certificate rooted from the federation root CA certificate. This is a technique used

<sup>11</sup> This problem will be eliminated with version of SAML 2.0 which specifically addresses the problem of SAML and XACML compatibility.

<sup>12</sup> <http://www.opensaml.org>

<sup>13</sup> <http://ws.apache.org/ws-fx/wss4j/>

by InCommon Federation to avoid administrative work for issuing certificates to its members.

As shown in Figure 1 ECL IdP is bundled with a CA, which can be configured to issue certificates to the federation's trusted members. ECL-SI also comes with utilities for repository to automatically obtain certificate from ECL CA (step 1) and publish this certificate to ECL registry (step 3). The live time of the certificate can be shorter than a commercially issued certificate. Hence, the security can be much stronger.

The users or the developers of client applications must provide the URLs of an ECL CA and an ECL AA to ECL-SI connector. If using a CA or an AA other than ECL CA or ECL AA, application developers must provide their implementation for connecting to their configured CA server or AA. On the other hand, ECL AA can accept authentication certificate issued by any CA as long as the root certificate of the CA is in the trust store and the certificate contains the user ID in certificate's UID field or Shibboleth crypto-handle in certificate's DN field<sup>14</sup>. In other word, a federation can choose to use any CA server with ECL AA.

All the processes are automated and hidden from the users. The users are only aware that ECL-SI will authenticate them to their CA server and retrieve attributes from their AA to allow them accessing repositories they want to access. Figure 1 shows ECL-SI client authenticates and obtains identity and opaque certificates from a designated CA (step 4). The certificates will be used for the entire session. The credentials can also be made available to the client prior or during its instantiation. When a user tries to access a repository, ECL-SI retrieves up-to-date information about the repository from the registry (step 5), determines if the user has all the security requirements, and uses the user's identity certificate to retrieve SAML assertion containing the required attributes from the AA (step 6). Next, the client formulates the SOAP request and uses the user's opaque certificate to access a resource on the repository (step 7).

On the repository side, the SAML assertion is extracted from the SOAP header; the message is decrypted (if necessary); and the assertion is validated against the certificate used for SSL connection, the expiry time and trusted IdP list. If the request passes all security checks, ECL-SI puts SAML assertion into the message context and forwards the message to repository implementation.

The last security check is the access control, which is handled by the repository. XACML policy engine or

other mean can be used to determine if a user can access a particular resource (step 8). Using XACML is an advantage if XACML policies are developed around user attributes as specified in the trust federation.

The ECL security solution has been deployed in the Course Management System at the Simon Fraser University. The infrastructure deployment includes installation of Shibboleth IdP with three specific plugins: certification authority, attribute authority and impersonation service. The CA plugin is integrated directly with the university LDAP server and authenticates the users via LDAP bind method. Our previous implementation utilized Centralized Authentication System (CAS) deployed at SFU. The Attribute Authority is integrated with the SFU LDAP server where it retrieves user's attributes. It is also registered in the InQueue Federation and the attribute assertions are trusted within the federation.

The CMS was configured as a Service Provider responding to the web services requests as specified in the ECL federated security profile (section 5.4). The access control decision are handled by the XACML policies generated from the access control data in the database (section 5.5) with respect to the attributes values for the users from the trusted organizations<sup>15</sup>.

## 7. Discussion and Related Work

The ECL security infrastructure has been deployed at Simon Fraser University and can provide certificates and signed attribute assertions to SFU users. The Course Management System has been made accessible via the web service with access control decisions made based on the user attributes and federated trust. As a result, ECL users from the trusted organizations can access resources with attributes issued and signed by their own institutions.

To the best of our knowledge our solution to providing access via web service with federated attribute-based security is novel and similar work does not exist. However, in different aspects, our work is closely related to several initiatives including Shibboleth, the WSS group at OASIS and the Liberty Project. Our approach extends Shibboleth from the Web environment into the Web Service environment. The WSS group concentrates on specifications, of which we make use. The Liberty Project [3] is much broader in scope than our work. In that respect, our work can be considered lightweight with the main

---

<sup>14</sup> Shibboleth crypto-handle is currently used by LionShare SASL-CA server. For more details see <http://lionshare.its.psu.edu>

---

<sup>15</sup> Although it is possible to directly inherit the trust from the federation we opted for manually pre-selecting the trusted institutions based on the federation data.

focus on easy setup of new trust networks and the low threshold enrollment of the new nodes into the trust network. Another difference is that the Liberty Project makes available only the specifications while the implementation is typically not available as it is done by the commercial partners of the project. Finally, there are two other projects that attract our attention: GridShib<sup>16</sup> (a solution for multiple organizations that wish to form a Virtual Organization or Grid) and MAMS. In the GridShib model, a SP can authenticate users (grid clients) using GridLogon and request additional users' attributes from Shibboleth AA (pull mode). In a push mode, which is similar to our approach, GridShib users authenticate and obtain attributes from Shibboleth AA and use them to make the request to the SPs.

Another interesting project utilizing XACML standards is MAMS Project [10] at Macquarie University. The project implements an inter-institutional authentication and authorization regime based on attribute exchange and XACML policies. However, the MAMS project is focusing purely on the Web environment.

## 8. Conclusions

In this paper we have presented the security solution supporting access to the repository base on the user attributes and trust federation. The solution defines security profiles, infrastructure services, and middleware components for low-barrier adoption by existing repositories. Although this infrastructure can scale to large networks it is particularly sensitive to the needs of medium and small organizations, which have complex attributes and accessing policies.

The infrastructure uses WSS standards. It is compatible with Shibboleth infrastructure for Web applications which enables developers to share infrastructure components such as attribute authority.

The infrastructure has been deployed and integrated with the authentication infrastructure at our university and interlinked with other institutions through the trust federation. We have enabled users from the trusted institutions with access to the repository of learning material with complex intellectual property arrangements for individual resources. The access policies in the repository were based on the user attributes. The pilot successfully demonstrated the federated security for users from two different organizations.

## 9. Acknowledgements

This work was partly supported by the LionShare project funded by the A.W. Mellon Foundation and LORNET Research Network funded by NSERC Canada.

## References

- [1] A. Nadalin, C. Kaler, P. Hallam-Baker and R. Monzillo., "OASIS Web Services Security: SOAP Message Security 1.0 (WS-Security 2004)," Tech. Rep. OASIS Standard 200401, March 2004.
- [2] R.L. Morgan, S. Cantor, S. Carmody, W. Hoehn and K. Klingenstein, "Federated Security: The Shibboleth Approach," *Educause Quarterly*, pp. 12-17, 2004.
- [3] J. Tourzan and Y. Koga, "Liberty ID-WSF Architecture Overview. Version 1.0," Liberty Alliance Project., 2004.
- [4] T. Scavo, "Shibboleth Architecture: Technical Overview," Working Draft 01, January 9, 2005, <http://shibboleth.internet2.edu/docs/draft-scavo-shib-techoverview-01.pdf>.
- [5] M. Hatala, G. Richards, T. Eap and J. Willms, "The interoperability of learning object repositories and services: standards, implementations and lessons learned," in *Proceedings of the 13th international World Wide Web conference, 2004*, pp. 19-27.
- [6] S. Carmody, M. Erdos, K. Hazelton, W. Hoehn, B. Morgan, T. Scavo and D. Wasley, "InCommon Technical Requirements and Information," vol. 2005, pp. 5, May 5, 2005.
- [7] D.F. Ferraiolo, D.R. Kuhn, and R. Chandramouli, "Role-Based Access Control", Artech House, 2003
- [8] R. Cover, Extensible Access Control Markup Language (XACML), Cover Pages page on XACML. Updated regularly. Available at <http://xml.coverpages.org/xacml.html>.
- [9] E.R. Weippl, "Security in E-learning", Springer, New York, 2005
- [10] J. Dalziel, E Wullings, "MAMS and middleware: The easily solved authentication, authorisation, identity, single-sign-on, federation, trust, security, digital rights and automated access policy cluster of problems", *Educause Australasia, Auckland, New Zealand*, April 5-8, 2005.

---

<sup>16</sup> <http://grid.ncsa.uiuc.edu/GridShib/>