

# Secure Communication Infrastructure for Object Repositories and 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 present the design and implementation of a lightweight security infrastructure, for the federated security, that enables organization to share resources based on the trust federation between several organizations. The infrastructure consists of an augmented security layer placed on top of the Web Service protocol. The solution utilizes the latest WS-Security specifications and, at the infrastructure level, is compatible with Shibboleth – a federated security solution for Web resources. In order to illustrate the potential of the infrastructure, we also describe the pilot application: an object repository with complex access policies.

## 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 multiple 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 background on federated security. In Section 3 we describe our security infrastructure while in Section 4 we present security profiles and analyze their security implications. Section 5 provides an account on implementation and describes the pilot case study. We conclude with discussion in Section 6 and outline future research in Section 7.

## 2. Security Standards and Initiatives

The Web Services Security (WSS) standard [1] defines how to send secured SOAP<sup>1</sup> messages over an insecure HTTP transport by embedding signatures and encrypted security tokens in the SOAP header. WS-Security Policy [3] is another set of specifications that provide a standard format for specifying how the implementation of Web Services can construct and check security headers. There is a substantial body of research results available in the verification of the security policies [4-6], and generating implementation code based on policies and abstract protocol descriptions [7-9], as well as tools for generating policies [6, 10].

Federated security aims to provide identity management and secure access to resources across multiple organizations [11]. The Liberty Project<sup>2</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 [12]. 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 [13].

Shibboleth<sup>3</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 [11].

In a SSO model, users sign onto their respective IdPs and can access all SPs in the federation. A federation is 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 [15]. 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>4</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.

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 and use it to

---

<sup>1</sup> <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>

<sup>2</sup> <http://www.projectliberty.org>

<sup>3</sup> <http://shibboleth.internet2.edu>

<sup>4</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 [15].

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.

### **3. Security Layer for Interoperability Infrastructure**

In this section we describe the design of the security communication layer as an extension of the eduSource Communication Layer (ECL) for connecting learning object repositories and learning services. The security layer for ECL extends the repositories' ability to distribute not only free material but also the material that has a certain level of access control restrictions.

#### *3.1 ECL*

ECL is a Web Services based middleware infrastructure that connects learning object repositories and services throughout Canada, USA, UK, Australia, and Europe [14]. 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 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 attribute authority. In fact, the solution is independent of the ECL protocol as it utilizes the SOAP messages header and leaves original ECL protocol messages unchanged. We will describe the ECL security design in the following sections.

#### *3.2. Security Infrastructure*

The ECL Security Infrastructure (ECL-SI) is a lightweight integrated solution. It builds on the idea of federation as introduced by the Shibboleth project. It is flexible in the sense that it enables repositories to join the federation with the minimal effort. Technically, the ECL-SI allows repositories to use any IdP and form their own circle of trust (federation). To provide this flexibility, the ECL-SI assumes that the repositories determine their own trust federation and access control policies while the ECL-SI provides the basic mechanisms and infrastructure components to support this trust. The ECL-SI is designed for Web Services and provides three components (Figure 1): Certification Authority (CA), Local Attribute Authority (LAA), and Service Registry (SR).

#### *3.3 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 ECL CA can be integrated with a standard login module, such as Kerberos (<http://web.mit.edu/kerberos/>), or a simple login with username and password. However, the infrastructure also allows both repository and application developers to load and use their own login modules.

### *3.4 Local Attribute Authority*

LAA is a specific term for AA. It is an attribute authority of an organization. Users obtain their required attributes assertions from their LAA to access resources at a service provider. A LAA can be coupled with a certification authority to avoid sending authentication to two different sites. The ECL LAA is a modified Shibboleth AA, which is adapted to work with Web Services. Assuming that a user's application knows the required attributes to access the service, the application formulates a Shibboleth attribute query request and calls LAA. The LAA is integrated with the organizational directory service where it obtains user attributes. Typically, the user certificate is also included into the SAML assertion and then the whole assertion is signed by the LAA to guarantee that the attributes are presented by their owner (so called holder-of-the-key method<sup>5</sup>).

### *3.5 Service Registry*

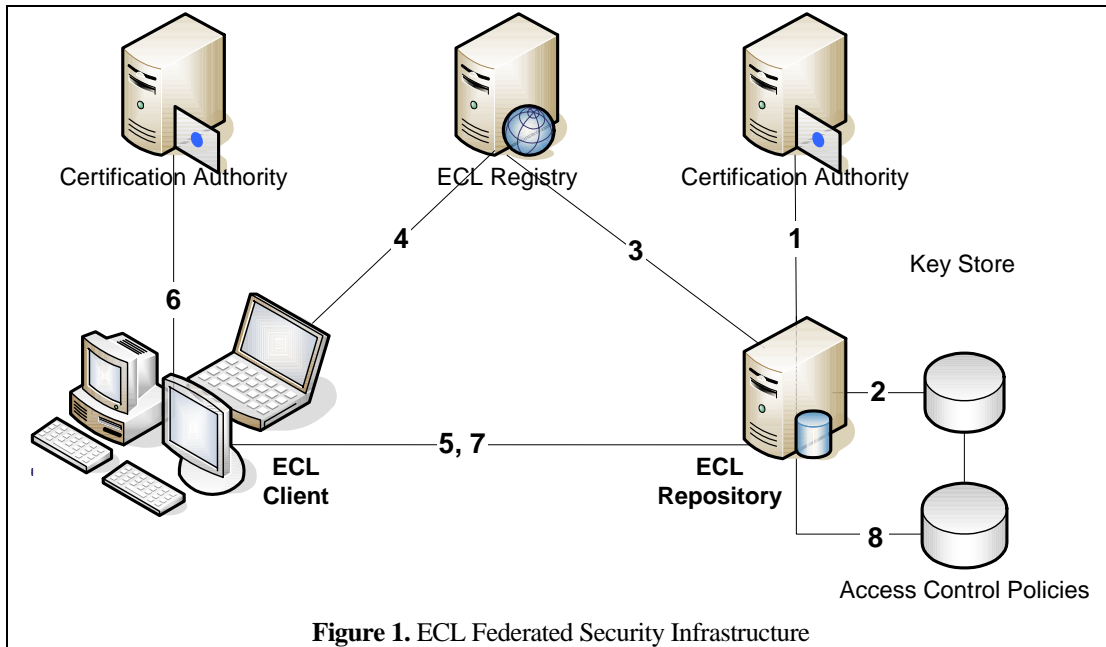
A service registry (SR) is a component that makes the infrastructure highly flexible and intelligent. The ECL SR holds information about service providers, concerning its supported security profiles, certificates, trusted LAAs, and which attributes are needed to access resources. Based on the information, the user's application can determine whether it supports the implemented secure communication method for a particular service provider. To access a resource, a user application begins by determining if the service provider is a trusted member of the federation, and then checks if its LAA is in the list of the service provider's LAA<sup>6</sup>. Finally, it checks if it has all the required attributes and the values requested by the service provider. 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.

In the ECL scheme, users do not have to know the SPs upfront. They can discover the SPs on the SR. This is different from the other federations such as InCommon [15] where the federation maintains the list of trusted entities and the list has to be distributed to all members of the federation.

---

<sup>5</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.

<sup>6</sup> This requirement can be relaxed by making the user directly responsible for the release of his/her attributes.



#### 4. Security profiles

A security profile is a description of a secure communication exchange between two parties. It is a messaging protocol between client applications and the service provider. 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, the encryption, and signature instructions of the SOAP 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

We describe both profiles in the following sections. The technical details can be found at the ECL website (<http://ecl.iat.sfu.ca>).

##### 4.1 Repository Controlled Security Profile

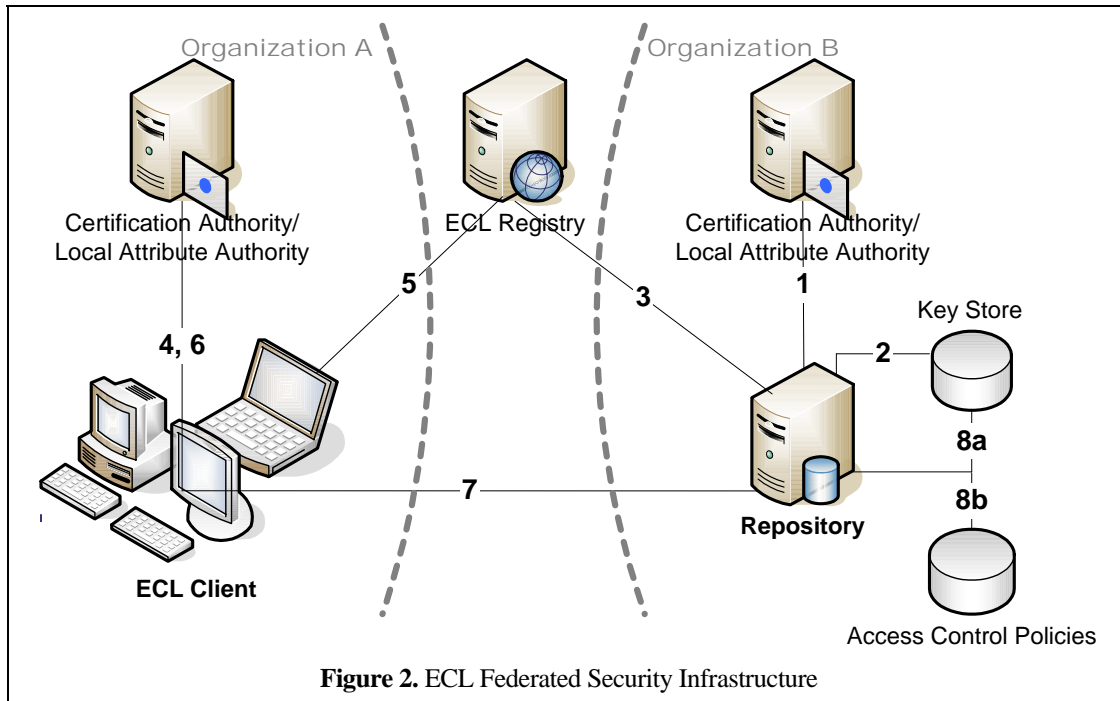
In the repository controlled security profile the user has to have an account at the target repository. The repository manages its own user base and users are responsible to negotiate with the repository for their access on individual basis. This typically means registering with the repository either for free or paid service. As a result, users obtain the username and password they use to authenticate themselves to the repository.

Figure 1 shows the flow of the messages and user's actions in the repository controlled profile. The user discovers the repository in the ECL registry and user clients obtains the necessary

information for the profile. To obtain the user name and password the information the registry record contains URL where the users can register with the repository. The ECL-SI includes the username and password into ECL messages with appropriate encryption. When the repository receives the request it validates the security parameters and grants access to the user.

Upon installation the ECL repository authenticates with its Certification Authority (CA) and obtains its certificate (Figure 1, step 1). Repository stores the certificate in its key store (2) and registers/updates its information including the certificate in the ECL Service Registry (3). ECL client retrieves the information about the repository from the ECL Service Registry (4) either after user searches the registry or from the client's pre-configured list of the repositories. The information contains the repository certificate, URL for obtaining user account on the repository, and technical parameters to establish the communication. If the user does not have an account with the repository it follows the URL and obtains the user account name and password (5). If the communication should be encrypted the user obtains his/her certificate from the generally trusted certification authority (6). Self-signed certificates can be used with the repository that does not require signed certificates. The username, password, and optionally user certificate are included into the ECL message and sent to the ECL repository (7). The message can be optionally encrypted so only the repository can decrypt the message. The repository authenticates the user against its access control policies (8). The returned results can be encrypted so only the originator of the request can decrypt it.

If the repository requires the encryption and signature, the ECL client uses repository's certificate to encrypt the message, and its own certificate to sign the message.



#### 4.2 Federated security profile

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

Figure 2 shows the communication flow of messages and information passed between a client, a repository and infrastructure services. The clients discover the services via the service registry. The users authenticate and request their security attributes from their home institutions. The repository validates the attributes and makes an authorization decision based on the users' attributes values 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 the 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 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. Implementation and Case Study

The ECL-SI connector is a middleware component that encapsulates and abstracts Web Service implementation from object repository developers. The implementation utilizes mainstream Java

Web Service implementations such as Axis SOAP engine, Shibboleth, OpenSAML<sup>7</sup>, and WSS4J<sup>8</sup>. We describe the process supported by the connector for the federated security profile.

To initiate security implementation, the ECL-SI would require basic information such as URLs of a CA and LAA, user authentication module (using ECL-SI default or a user plug-in module), and user credentials needed to authenticate to the user's CA and LAA. When the ECL-SI connector is loaded at the object repository in organization B, it undergoes step 1, 2, and 3 in Figure 2 consecutively. It begins by loading security configurations, generating key pairs, and sending public key to be certified by the CA. The certificate and the keys are then stored into the keystore. Finally, the connector has all the information to deploy and register the service to the SR. Note that the list of required attributes and the list of trusted LAAs and CAs must be located in the configuration when registering the service onto the SR.

At the client application in Organization A, the ECL-SI connector would use a similar process to instantiate a security profile (step 4). The credentials can be made available to the connector at the same time a new repository service is added. However, before adding a new repository service, the users should check with the ECL-SI connector if they have necessary security clearance to access the repository's resources. Assuming that the users have security clearances, step 6, 7, and 8 are processed automatically by the ECL-SI connectors. The connector retrieves the necessary user's attributes from the designated LAA. The attributes are asserted by the LAA for a specific repository and are issued in SAML attribute assertion format. The assertion is inserted into the SOAP header using WSS4J API utilities. When the SOAP message reaches the SP repository, WSS4J strips the security token in the SOAP header, decrypts the message, validates the message's signature (if necessary), and passes the security tokens to the ECL-SI connectors. Finally, the ECL-SI loads the authorization module to validate the security tokens and passes the tokens to the access control policy module to make the final decision on the security clearance.

### *5.1 Case Study: Course Management Systems*

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>9</sup>. The learning objects include material developed by faculty (documents, Web pages, media documents, applets, etc.); 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 agreements with publishers; and material directly purchased from publishers for specific course offerings. 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. Although access to these resources can differ based on the user roles such as faculty-at-sfu, faculty-at-canadian-university, general-public, undergraduate-registered-to-itec426-fall2005, etc., the current solution conservatively locks the

---

<sup>7</sup> <http://www.opensaml.org>

<sup>8</sup> <http://ws.apache.org/ws-fx/wss4j/>

<sup>9</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>).

whole repository and only students and faculty directly associated with the courses can access the material after they log in to the system.

To broaden the access to the CMS for all academic community and public we have implemented ECL web-service with the federated security profile. The clients can now search and retrieve objects in CMS based on the user attributes signed by their home institutions that are part of the federation. The ECL request to CMS includes the attributes in the SAML format. After all security checks on the consistency of the message and validity of the signatures the attributes are matched against the XACML policies and authorization decision is made. This allows CMS to return the maximum number of objects that the user is entitled an access to.

## **6. 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. A trust network has been created with PennState University. As a result, ECL users at both SFU and PennState can access resources in the SFU Course Management System using the attributes issued by their own institutions.

To the best of our knowledge 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 [12] is much broader in scope than our work. In that respect, our work can be considered lightweight with the main 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>10</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. MAMS Project<sup>11</sup>, at Macquarie University, 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.

## **7. Conclusions and Future Research**

In this paper we have presented the security infrastructure for web services. The infrastructure provides the security framework for object repositories to create trust federation as well as to provide services with different levels of security. The solution defines security profiles, infrastructure services, and middleware component 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.

---

<sup>10</sup> <http://grid.ncsa.uiuc.edu/GridShib/>

<sup>11</sup> <http://www.melcoe.mq.edu.au/projects/MAMS/index.htm>

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 in a pilot application to provide access to a 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.

Our current research focuses on bridging between our solution and attribute-based security that is being developed for the peer-to-peer network within the LionShare project.

## 8. 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. We would also like to thank Steven Carmody and all Shibboleth working group for their input to defining the profiles.

## 9. 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.
- [3] G. Della-Libera, P. Hallam-Baker, M. Hondo, T. Janczuk, C. Kaler, H. Maruyama, N. Nagaratnam, A. Nash, R. Philpott, H. Prafullchandra, J. Shewchuk, E. Waingold and R. Zolfonoon, "Web services security policy language (WS-SecurityPolicy)," December 2002.
- [4] J.D. Guttman and A.L. Herzog, "Rigorous automated network security management," *International Journal of Information Security*, vol. 4, pp. 29-48, February 2005.
- [5] A.D. Gordon and R. Pucella, "Validating a Web service security abstraction by typing," in *XMLSEC '02: Proceedings of the 2002 ACM workshop on XML security*, 2002, pp. 18-29.
- [6] K. Bhargavan, C. Fournet and A.D. Gordon, "Verifying policy-based security for web services," in *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security*, 2004, pp. 268-277.
- [7] D. Song, A. Perrig and D.E.-. Phan, *AGVI - Automatic Generation, Verification, and Implementation of Security Protocols*, 2001.
- [8] S. Lukell and A. Hutchison, "Automated Attack Analysis and Code Generation in a Multi-Dimensional Security Protocol Engineering Framework," in *Proceedings Southern African Telecommunications Networks and Applications Conference 2003*, 2003,
- [9] D. Pozza, R. Sisto and L. Durante, "Spi2Java: Automatic Cryptographic Protocol Java Code Generation from spi calculus," in *AINA '04: Proceedings of the 18th International Conference on Advanced Information Networking and Applications Volume 2*, 2004, pp. 400.
- [10] M. Tatsubori, T. Imamura and Y. Nakamura, "Best-Practice Patterns and Tool Support for Configuring Secure Web Services Messaging," in *ICWS '04: Proceedings of the IEEE International Conference on Web Services (ICWS'04)*, 2004, pp. 244.
- [11] R.L. Morgan, S. Cantor, S. Carmody, W. Hoehn and K. Klingenstein, "Federated Security: The Shibboleth Approach," *Educause Quaterly*, pp. 12-17, 2004.
- [12] J. Tourzan and Y. Koga, "Liberty ID-WSF Architecture Overview. Version 1.0," Liberty Alliance Project., 2004.
- [13] T. Scavo, "Shibboleth Architecture: Technical Overview," Working Draft 01, January 9, 2005, <http://shibboleth.internet2.edu/docs/draft-scavo-shib-techoverview-01.pdf>.
- [14] M. Hatala, G. Richards, T. Eap and J. Wilms, "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.
- [15] 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.