Implementing a Secure Annotation Service

Imran Khan, Ronald Schroeter, Jane Hunter

The School of ITEE
The University of Queensland,
St Lucia, Queensland, Australia
{imrank, jane}@itee.uq.edu.au

Abstract: Annotation systems enable "value-adding" to digital resources by the attachment of additional data in the form of comments, explanations, references, reviews, corrections and other types of external, subjective remarks. They facilitate group discourse and capture collective intelligence by enabling communities to attach and share their views on particular data and documents accessible over the Web. Annotation systems vary greatly with regard to the types of content they annotate, the extent of collaboration and sharing they allow and the communities which they serve. However within many applications, there is a need to restrict access to the annotations to a particular group of trusted users - in order to protect intellectual property rights or personal privacy. This paper describes a secure, open source annotation system that we have developed that uses *Shibboleth* and *XACML* to identify and authenticate users and restrict their access to annotations stored on an *Annotea* server.

1 Introduction

Annotations have long been used as a a tool to facilitate collaborative scholarly discourse. They enable users to attach additional material such as comments, notes, queries, assessments, references to resources such as documents, images or datasets. When applied to digital resources shared via the Web, they provide a very powerful collaborative tool - enabling the easy capture and wide dissemination of individuals' and group opinions of particular digital resources.

Currently available annotation systems vary widely with respect to the types of content they annotate, the extent of collaboration and sharing they allow and the communities which they serve [1]. Although they have been successfuly applied to domains including education [8-9], research, medicine [10] and neuroscience [11] to enable the capture and exchange of metadata, ideas, opinions and interpretations, evaluation of these applications indicates limitations in existing commercial and prototype systems. Current systems are limited by: lack of responsiveness, use of non-standard proprietary technologies; lack of authentication of the annotations' creator; limited search capabilities; lack of security mechanisms; inability to reply to/stagger annotations; asynchronous sharing only ; limited media types supported; coarse granularity of the annotations; unstructured annotations (single field, free text only).

The main focus of the work described within this paper is to provide annotation tools for collaborators within eResearch environments – and particularly higher education environments. A critical requirement for such a domain is the need to be able to restrict access to annotations

attached to a particular collection of digital resources, to a particular group of trusted colleagues - for reasons of privacy, confidentiality or protection of intellectual property. Particularly in eScience, the annotation or interpretation of the raw document or data, is often more valuable than the resource it annotates. Also by providing researchers with a robust, reliable security infrastructure, they may be more willing to engage in the exchange of views and ideas – a key to successful inter-organizational collaboration.

The security requirements for annotations involve two levels of protection:
1. protecting the annotation server on which the annotations are stored, through some form of authentication [3,4]
2. authenticating and protecting the individual annotations through the specification of access policies defining permissable types of access by individual users or user types (e.g, list, create, read, edit, delete) based on user attributes.

Within this paper we describe an open source implementation of a secure annotation service that we have developed. Our implementation involves the combination and extension of a number of existing open source technologies that are currently available and use open standards:
- Annotea – a Web-based annotation server developed by the W3C as part of the Semantic Web initiative [16];
- Shibboleth – an Internet2 middleware initiative that enables identity management and secure access to Web resources shared amongst multiple organizations [5];
- XACML (eXtensible Access Control Markup Language) – XML-based language for defining and enforcing access control policies [25].

The remainder of this paper describes in more detail the secure annotation system that we have built. The paper is structured as follows:
- Section 2 describes previous related activities in the development of annotation systems and security mechanisms;
- Section 3 describes the overall architecture of our system and its main components:
- Section 4 illustrates the user interface and system's functionality;
- Section 5 provides an evaluation of the system to date, our conclusions and describes future plans for this research.

2 Background and Previous Work

Significant prior work has been carried out on both web-based annotation systems and on identity management and role-based access control. Rather than re-invent the wheel, we carried out an analysis of existing systems to determine if any currently available solutions satisfied our needs and hence could be integrated, refined or extended.

2.1 Existing Annotation systems and Annotea

A survey of current Web-based annotation systems [1] reveals that they vary in the way in which annotations may be attached, the way in which they are presented and in the access control mechanisms. Some systems are designed for private use only whilst others permit sharing amongst groups and/or public access. None of the surveyed systems provide the kinds of fine-grained access control mechanisms that is required by collaborative teams of scientists engaging in eResearch.

Through earlier work [3] we identified Annotea [16] as an ideal approach for implementing an annotation server. Annotea is a Web-based annotation system that uses Resource Description Framework (RDF) [14] to model annotations as a set of statements or assertions made by the author. These annotations are then stored in a HTTP enabled server, which enables clients such as Annozilla [24] and Amaya [23] to query, update, post, delete and reply to annotations. Currently there are two publicly available implementations of annotation servers which use Annotea: Zope [3] and W3C Perllib [4].
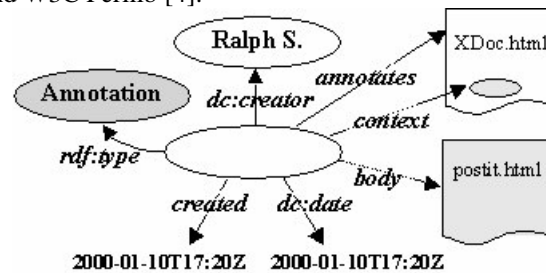
Figure 1: Koivunen et al Basic Annotation Schema [16]

A key strength of the Annotea protocol is that it uses open W3C standards such as RDF, XPointer, XLink and HTTP. Figure 1 illustrates the RDF annotation schema used to describe various properties of an annotation including its *author*, *title*, *date of creation, body and context*. XPointer technology is used to point to a specific location within a structured Web document and thus describe the context of an annotation. By choosing to use RDF, Annotea makes it possible to easily adapt or extend the existing scheme to incorporate additional information (e.g. what type of annotation it is, what language it is in, the type of resource that it can annotate, structured annotations). The use of machine-processable RDF descriptions also enables easier search, retrieval and linking of the annotations to related resources and services using semantic web technologies (e.g., OWL, SWRL). Annotea can also be easily extended to allow for the annotation of media types other than text e.g., images through the use of SVG [15]. Vannotea [2] uses a similar approach to extend Annotea to enable the annotation of videos. These applications and in particular the application of Vannotea to Indigenous Knowledge Management [12] clearly identified the need to extend the Annotea server to enable fine-grained access control to the annotations.

2.2 Identity Management and Shibboleth

Harris et. al. generated a comprehensive report describing AM systems used in the UK Higher Education sector [7]. The most prominent systems identified included: Microsoft's Passport, Liberty Alliance, PAPI, WS-Security, Shibboleth and Athens. Of the six prominent systems, only three fall into the category of systems targeted at the HE domain, while the other three (Passport, Liberty Alliance and WS-Security) are primarily focused on providing a business-centric solution. At present in the UK, the major AM solution is Athens, a system developed by the UK HE community. Its key distinction is that it uses a single centralized database which maintains a list of Athens username/passwords for all users with accounts at participating institutions. Although Athens uses distributed administration and physical database replication, it does not exhibit true Single Sign-on capability. The users' username/password must be re-entered for each new resource accessed.

The shortcomings of Athens lead to the emergence of projects such as Shibboleth. Morgan et al [17] describe Shibboleth as "an open-source system that extends Web-based applications and identity management for secure access to resources among multiple organizations." Shibboleth is based upon a number of open standard protocols including HTTP, XML, XML Schema, XML Signature, SOAP (Simple Object Access Protocol). In particular it is dependent on:

- SAML [19] Security Assertion Markup Language for the exchange of assertions between the Identity Providers and Service Providers
- eduPerson [26] – an EDUCAUSE initiative to define a standard set of person attributes in higher education environments

Figure 2 illustrates a simplified view of a Shibboleth transaction. The two key entities in this model are the Service Provider (SP)and the Identity Provider (IDP). Shibboleth is concerned with securely transferring attributes from the Identity Provider to the Service Provider so that an authoritative decision can be made. When a user attempts to access a Web resource, a process called the SHAR (Shibboleth Attribute Requestor) intercepts the request. The SHAR then interacts with a process on the IDP called the AA (Attribute Authority) which returns attributes about the user which made the request. The SHAR can then pass the attributes on to a Resource Manager on the SP which is responsible for making access control decisions. All of this communication uses SAML assertions over HTTPS as a secure way of transferring the attributes.
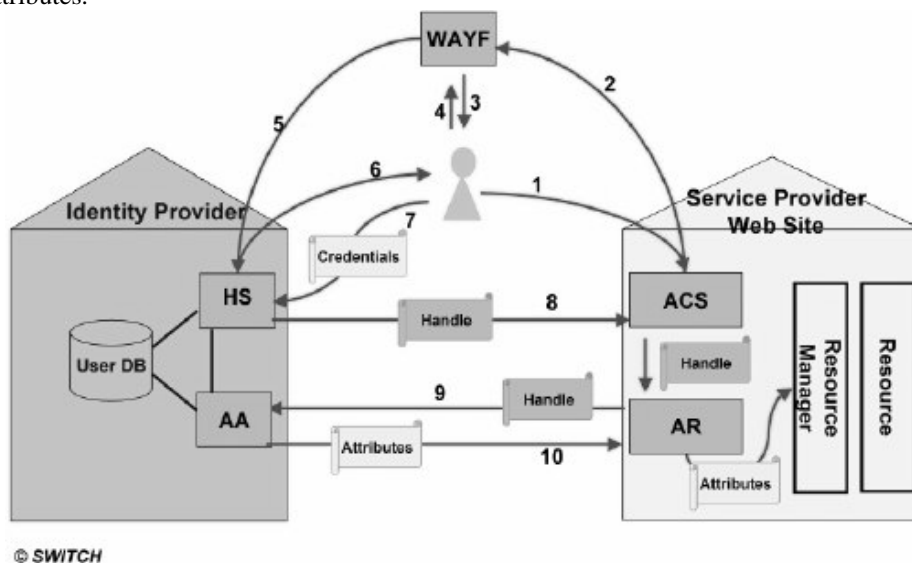

Figure 2: Shibboleth-Enabled Transaction [5]

2.3 XACML

As Lorch et al [18] explains, Shibboleth does not provide a dynamic and distributed approach to access control. XACML enables us to address these issues. XACML (eXtensible Access Control Markup language) is an XML-based language used to describe general purpose access control policies as well as access control decision request/response language [25]. XACML is composed primarily of 2 main language constructs: a) the syntax for defining the language and b) the semantics for processing these policies. XACML policies are expressed in XML and must

conform to an XML schema that defines the language semantics. Policies are defined in a tree-like structure as a set of rules pertaining to a particular resource and subject. The second language construct of XACML are the requests and the responses to these requests, both of which are also expressed as XML. Each request is composed of attributes associated with the requesting user, the resource being acted upon, the action being performed on the resource and environmental information. The response can be one of four specific types: Permit, Deny, Not Applicable, and Intermediate. We have chosen to use XACML within this project to define policies for access control of the annotation server. Specifically we will be using Sun's XACML implementation [29] which includes an XACML engine and necessary tools for its integration into our annotation server.

3   System Architecture and Implementation

Figure 3 illustrates the overall architecture of the system. The diagram highlights the two key components of the Shibboleth architecture: Identity Provider (IDP) and Service Provider (SP).
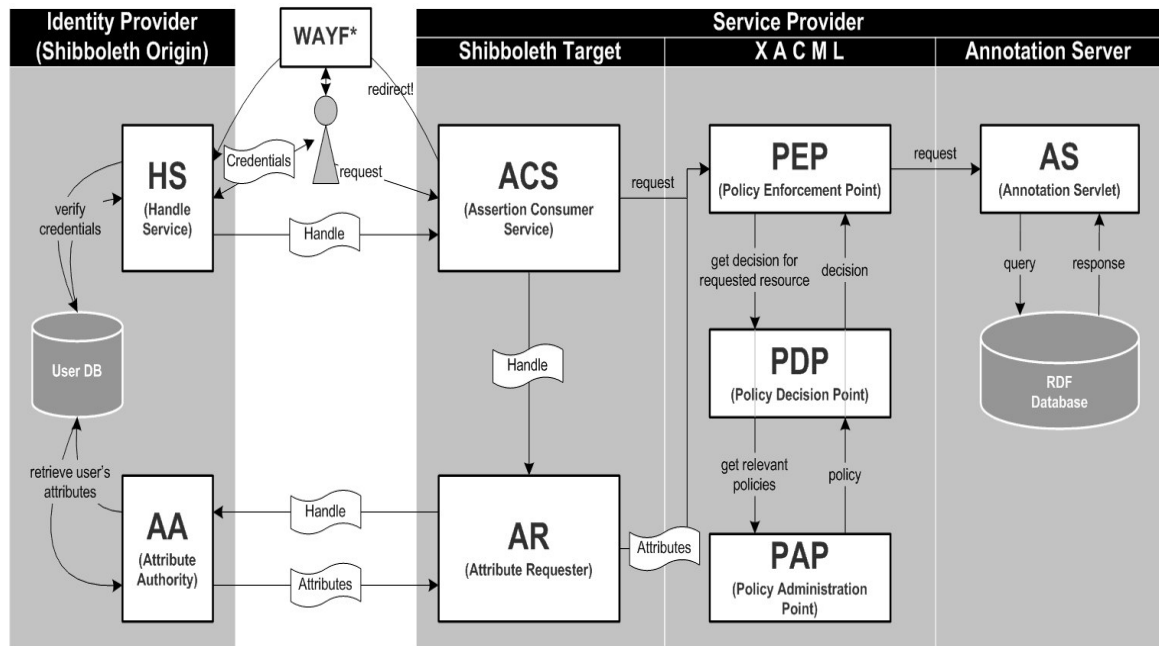
Figure 3: System Architecture

The annotation server within Figure 3 is part of the Shibboleth SP, and may be located on any of the organizations/universities that are part of the federation. The organization which hosts the annotation server is responsible for configuring the Shibboleth access control for the annotation server so that it is shared with the other members of the federation. It is the responsibility of the other universities to configure their IDP so that its users can gain access to the annotation server and also provide the relevant authentication measures as agreed by the different universities (circle of trust) as well as the process defined by the Shibboleth identity management system.

3.1 Client-side

The user's client side application is responsible for the user interface that enables:
- the retrieval and display of annotated web resources;
- display, search and browsing of annotations (e.g. through a browser plugin or side-bar);
- creation, editing, deletion and attachment of annotations to Web resources;
- creation, editing, deletion and attachment of access policies to annotations.

It also has to process and exchange information in compliance with the Annotea protocol. It is important to note that annotated objects may be outside the federation i.e., our system is not responsible for managing access control of the resources that are being annotated.

A number of annotation tools already exist for annotating Web resources. In particular we evaluated three potential client side annotation tools: Amaya [23], Annozilla [24], and Vannotea [2]. Although suitable for attaching textual or hyperlink annotations to digital objects or segments of those objects, and for viewing and browsing the annotations – none of these applications provided an interface suitable for also specifying XACML access policies and attaching them to the annotations.

Consequently we developed our own client-side application as an extension to Annozilla. The initial version of the client was developed using Ajax (Asynchronous JavaScript and XML). Our initial application runs on a Mozilla browser and uses the DOM API to download, display and edit both annotations and policies – and to map between the user interface and XACML policy definitions. More recently an additional client side application has been developed using C# and .Net which appears as a side-bar to the Internet Explorer browser (see Figure 6).

3.2 Server-side

 The Server side consists of four main architectural components: the Annotation server, XACML module; Shibboleth attributes and the Jena database.

3.2.1 The Annotation Server

This has been implemented using Java Servlets, hosted on a Tomcat server. In addition to the operations defined by Annotea (posting, querying, downloading, updating, replying and deletion of annotations) the server has been extended to support the fine-grained access policies required by this project. Figure 4 illustrates the extensions which we have made to support the inclusion of policies (in red). The first extension is the unique *creatorID* property of the annotation as well as to the body and policy objects. It provides a unique user id (*eduPersonPrincipalName*) to describe the creator of an annotation, body or policy. This user id must be unique both across, as well as within organizations. The creatorID property is used to make decisions on *delete* and *update* operations - only the creator of a resource is permitted to modify or delete that resource.

The other key extension is the policy object. This is an XACML policy referenced using a unique URL. Policies are stored within the RDF repository, along with annotation bodies. They can be created either during the posting of a new annotation or independently of an annotation. Annotations are linked to particular policies through their *policy* property – which specified by a URL. This approach has the benefit of enabling multiple annotations to use the same policy. If a policy is modified, the changes will effect all those annotations associated with the policy.
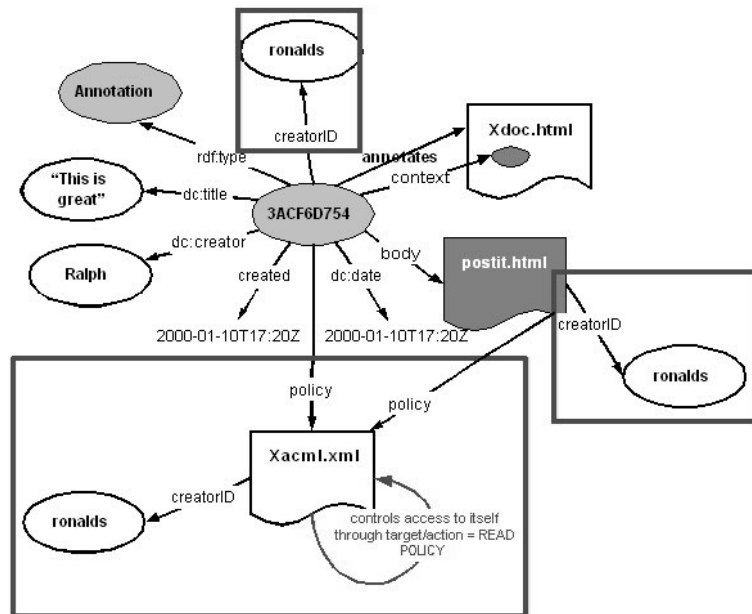
Figure 4: Extended Annotea Model

Additional methods required to support the policy-related operations included:

1. *addAnnotationPolicy*: given an annotation URL, a *creatorID* and an XML representation of the policy, this method adds the policy to the RDF repository.
2. *addPolicy*: given a *creatorID* and RDF representation of a policy, this method adds the policy into the repository. This method is called when a policy is being added independently of any annotation.
3. *updatePolicy*: given the URL of a policy and an XML representation of the policy itself this method removes the existing policy with the given URL and adds a new policy with the same URL.
4. *getPolicyByCreatorID*: given a unique *creatorID*, this method retrieves all policies that have been created by the given *creatorID*.

3.2.2 XACML Module

This module is responsible for implementing the Role Based Access Control functionality. It makes decisions on whether a particular request is permitted based upon the role/attributes of the person making the request. The creation and reply of annotations is open to all users that are permitted to access the annotation server. Updating and deleting of annotations and policies are operations which are only available to the creator of the annotation or policy. There are three types of actions permissible on annotations by users other than the creator:

- *LIST* – viewing of annotation metadata (e.g., author, creation date, language, etc.)
- *READ* – viewing of the annotation body.
- *READ_POLICY* – viewing of the annotation policy.

Figure 5 illustrates an example policy and example request. Each *Policy* consists of a set of *Rule*s related to whether a specific operation is permitted by a particular Subject. The Subject is

described by a set of attributes which identify the credentials of a particular user e.g. affiliation, role, etc. In Figure 5a, the Staff policy is characterized by having an attribute *eduPersonAffiliation* equal to "staff". This is a relatively simple example. It is possible to define groups using multiple attributes. In the example, staff are permitted to perform all three operations on annotations whilst students are denied access to all three. The Request is for a student to be permitted to read policy 123. Given the example policy, this request will be denied.

Figure 5: (a) Example Policy and (b) Example Request

The XACML module's Policy Decision Point (PDP) and Policy Enforcement Point (PEP) have been integrated into the annotation server through three basic operations. The first operation involves gathering attributes about the requester provided by the requestor's origin site through SAML assertions. Using these attributes it is possible to generate a XACML request based upon the action to be performed (e.g. **LIST, READ**, **READ_POLICY**), the resource requested and the attributes associated with the respective requestor or subject. The second part of the XACML module involves locating the policy associated with the resource being requested. This is simply a matter of querying the RDF repository to retrieve the policy of a particular annotation, given its URL. Once the policy is retrieved, it can be compared with the request in order to make a decision. Sun's XACML implementation [29] generates an appropriate XACML response specifying whether a request is permitted or not. If a request is not permitted (as in the above example), then a request denied message is presented to the user.

3.2.3 Shibboleth (SAML) Attribute Assertions

The annotation server depends on Shibboleth to provide the necessary eduPerson attributes about a requestor, as provided by its origin site (Identity Provider). These attributes are used by the XACML module to make an access control decision. Shibboleth itself is a complex architecture and further details are available from [27]. Each site which takes part in a Shibboleth federation may consist of either/both an origin (identity provider) and target (service provider). In terms of our annotation system, an annotators' attributes are provided by their origin (Identity Provider), which stores them in a directory service such as an LDAP server. The annotation server is hosted on the target site. This makes the server available to members of organizations that are part of the federation and have sufficient access privileges to the annotation server as defined by the host organization.

3.2.4 Jena Database

Jena [28] provides an API to an RDF repository and in the context of this system is responsible for enabling the storage and interfacing of data – this includes annotations, policies and annotation bodies, which are all stored in the repository as RDF instances. Jena itself sits on top of MySQL database which stores the RDF data as a relational database with a schema defined by Jena. The annotation server uses the Jena API to interact with data stored in the MySQL database. The Jena API also enables us to provide a search function over the annotations – on the creator, date, language, in_reply_to fields.

4 The User Interface

For testing and illustrative purposes, we used the ePrints archive at the University of Queensland. Figure 6 shows the user interface after a user with authenticated access to the annotation server logs onto the system and retrieves a particular annotated publication. The annotations are displayed in the top left-hand frame, the details of a selected annotation are in the bottom left-hand frame and the publication is displayed in the right hand frame.

Figure 6: User Interface showing threaded annotations and annotation metadata

Figure 7 illustrates the side-bar interface which was developed to define policies. It consists of two main parts; the definition of access control rights to a particular user group and the definition of user groups (based on particular eduPerson attributes).
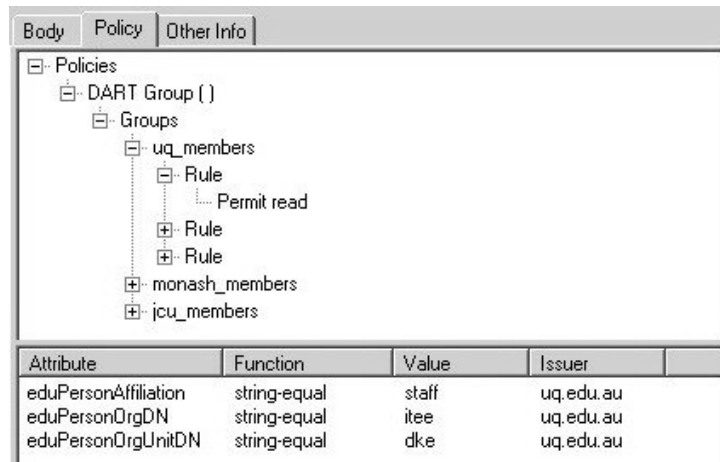
Figure 7: User Interface for defining access policies

Figure 8 illustrates the browser side-bar that provides the user interface for creating and attaching an annotation. We have extended Annotea to support structured annotations that contain a number of fields including hyperlinks, files, free text or controlled vocabularies.
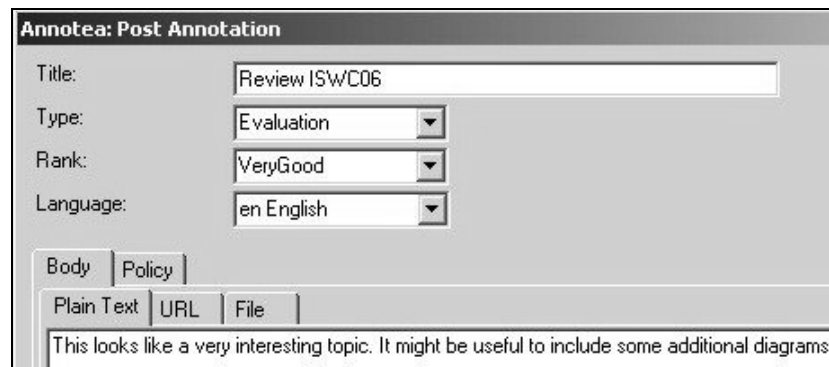

Figure 8: User Interface for creating/editing an annotation

5 System Evaluation, Future Work and Conclusions

5.1 System Evaluation

To date, system evaluation has consisted of unit and system testing and limited usability tests. A number of scenarios (involving the creation and updating of policies and annotations) were used to thoroughly test the system. We also tested the various policies by logging in as users with different attributes and by modifying the attributes directly in the LDAP directory. In all cases the annotation server behaved as expected – restricting access to policies by users in compliance with the rules.

Issues which did arise during the testing phase included:

- Current implementations of Annotea do not support queries on the content of the annotation – only on a number of annotation attributes (e.g. author, date, language)
- Allowing the deletion and update of annotations, leads to the problem of having 'hanging references' where replies refer to annotations which have either been updated or deleted.
- The use of URLs to identify policies enables them to be re-used and applied to multiple annotations. However this causes complications when deleting policies – due to the possible run-on effects when multiple annotations reference a single policy

- *Obligation*s – this XACML feature could be useful e.g, to define a condition that when an annotation has been replied to by someone, an email must be sent to the creator of the annotation to inform them of this event.
- Shibboleth allows users to define attribute release policies – that restrict the release of certain attributes for user privacy. This may be problematic if these attributes are included in the policy rules. Users may be denied access to annotations that they have a right to access.

5.2 Future Work

Aspects of this work that would benefit from further investigation include:

- User evaluation: detailed usability studies are required to acquire user feedback and determine functional requirements of different user groups; how intuitive, friendly and efficient the user interface is; and improvements, refinements and extensions to the system.
- Reduced reliance on Shibboleth: approaches other than Shibboleth and the *eduPerson* profile will enable the annotation server to be used outside of the Higher Ed sector.
- Annotation of PDF files: the popularity of publishing scholarly information in PDF format indicates increasing demand for the annotation of PDF documents. Extending Annotea in such a way would lead to the use of proprietary technology but has huge potential.
- Annotation of databases and spreadsheets: scientists in particular are under increasing pressure to publish their raw data sets with their journal publications. In parallel with this trend will be a growing requirement for the ability to annotate databases and spreadsheets.
- Access policies based on document attributes: the current system is based on policies associated with user attributes. It would be interesting to investigate policies that are based on attributes or characteristics of the digital resources.
- Complex querying: the integration of Algae or some other advanced querying method would allow more complex queries over the annotation server.
- Web browser extension: although Annotea compliant extensions exist for Web browsers, they do not provide support for policy definitions. It is necessary to either extend an existing Annotea plugin or develop a new extension.
- Non Web-based evaluation: to date the annotation server has only been tested on the annotation of HTML documents. However it will become necessary to evaluate the annotation of arbitrary document and media types (e.g., audio, video, image).
- Scalability: further investigation is required to determine how the system performs as the number of annotations, the number of access policies and the number of users grows? Faster alternatives to Jena may be required.
- Integration within Vannotea: Vannotea allows real-time collaboration of digital objects by distributed groups of users through video-conferencing and application sharing. The aim is to incorporate the work developed within this thesis for stand-alone asynchronous annotations, within the Vannotea system.

5.3 Conclusions

This paper describes a secure annotation service that we have developed based on a combination of existing open source technologies. Secure annotation servers are required in many domains including telemedicine, peer reviews and eResearch. By providing clinicians and researchers with the necessary support for maintaining confidentiality and protecting both intellectual property and personal privacy associated with their annotations, they will be more willing to share their views and engage in inter-organizational and inter-disciplinary collaborations. The modular design and use of interoperable technologies makes it possible to easily extend the server to support the annotation of additional resource types or to use it as a standard public annotation server.

References

[1]   R. M. Heck, S. M. Luebke, and C. H. Obermark, "A Survey of Web Annotation Systems", 2005, http://www.math.grin.edu/~luebke/Research/Summer1999/survey_paper.htm%.

[2]   R. Schroeter, J. Hunter, and D. Kosovic, "Vannotea - A Collaborative Video Indexing , Annotation and Discussion System For Broadband Networks," in Knowledge Markup and Semantic Annotation Workshop, K-CAP 2003, Sanibel, Florida, 2003.

[3]   "Zope Annotation Server", 2005, http://www.zope.org.

[4]   "W3C - Perllib Annotations Server", 2005, http://www.w3.org/1999/02/26-modules/User/Annotations-HOWTO.

[5]   "Shibboleth Project - Internet2 Middleware", 2005, http://shibboleth.internet2.edu/.

[6]   "EduServ Athens for education", 2005, http://www.athens.ac.uk/.

[7]   N. Harris, S. McLeish, and J. Paschoud, "Access Management Report," London School of Economics, Technical Report 2002.

[8]   E. Desmontils, C. Jacquin, and L. Simon, "Dinosys: An Annotation Tool for Web-Based Learning," Third International Conference ICWL, Beijing, China, 2004, pp. 59 - 66.

[9]   M. Kirsch-Pinheiro, M. R. S. Borges, and J. V. d. Lima, "A framework for awareness support in groupware systems," *Comput. Ind.*, vol. 52, pp. 47-57.

[10]   M. Lewkowicz, G. Lortal, A. Todirascu, M. Zacklad, and M.-F. Sriti, "A Web-based Annotation System for Improving Cooperation in a Care Network," in ICWE Workshops, 2004, pp. 227-239.

[11]   M. Gertz, K.-U. Sattler, F. Gorin, M. Hogarth, and J. Stone, "Annotating Scientific Images: A Concept-Based Approach," in 14th International Conference on Scientific and Statistical Database Management, 2002.

[12]   J. Hunter, R. Schroeter, B. Koopman, and M. Henderson, "Using the Semantic Grid to Build Bridges between Museums and Indigenous Communities," in GGF11 Semantic Grid Applications Workshop, Homolulu, 2004.

[13]   R. Schroeter, J. Hunter, and D. Kosovic, "FilmEd - Collaborative Video Indexing, Annotation and Discussion Over Broadband Networks," in 10th International Multimedia Modelling Conference, Brisbane, Australia, 2004.

[14]   "Resource Description Framework (RDF) / W3C Semantic Web Activity", 2005, http://www.w3.org/RDF.

[15]   M.-R. Koivunen, "Extending the Annotea addressing schema", 2003, http://www.w3.org/2001/Annotea/Plan/context/newcontext.html.

[16]   M.-R. Koivunen, R. Swick, E. Prud'hommeaux, and J. Kahan, "Annotea Protocols", 2002, http://www.w3.org/2001/Annotea/User/Protocol.html.

[17]   R. L. Morgan, S. Cantor, W. Hoehn, and K. Klingenstein, "Federated Security: The Shibboleth Approach," *EDUCAUSE Quarterly*, vol. 27, pp. 12-17, 2004.

[18]   M. Lorch, S. Proctor, R. Lepro, D. Kafura, and S. Shah, "First experiences using XACML for access control in distributed systems," in Proceedings of the 2003 ACM workshop on XML security Fairfax, Virginia 2003.

[19]   R. Cover, "Cover Pages: Security Assertion Markup Language (SAML)," OASIS, Technology Reports 2005.

[20]   Internet2, "Internet2 - Home," 2005. http://www.internet2.edu

[21]   M.-R. Koivunen, "Annotea Project", 2005, http://www.w3.org/2001/Annotea.

[22]   M.-R. Koivunen and J. Kahan, "Annotea: an open RDF infrastructure for shared Web annotations," in Proceedings of the 10th international conference on World Wide Web, Hong Kong, 2001

[23]   W3C, "Amaya Home Page," 2005. http://www.w3.org/Amaya

[24]   M. Wilson, "Annozilla (Annotea on Mozilla)," 2005. http://annozilla.mozdev.org

[25]   T. Moses, "eXtensible Access Control Markup Language (XACML) Version 2.0," OASIS, Specification Document Feb, 2005.

[26]   EDUCAUSE, "EDUCAUSE | Net@EDU | eduPerson Object Class ", 2005. http://www.educause.edu/eduperson

[27]   T. Scavo and S. Cantor, "Shibboleth Architecture," Internet2, Technical Overview June, 2005.

[28]   B. McBride, "Jena: A Semantic Web Toolkit," *IEEE Internet Computing*, vol. 6, pp. 55-59, 2002.

[29]   Sun Microsystems, "Sun's XACML Implementation," 2005. http://sunxacml.sourcefore.net