

Logins for Life



Leveraging Social Networks to Gain Access to Organisational Resources

D.W.Chadwick, M.S. Ferdous, G. Inman, K.W.S. Siu
University of Kent

Abstract

We describe a federated identity management system that allows users to access organisational resources using their existing login accounts at social networking and other sites. Users are empowered to link their various login accounts together, so that they can swap between them. We utilise the Level of Assurance (LoA) concept to ensure the organisation's site remains secure, and we allow users to escalate their login privileges by logging in via higher LoA accounts when necessary. The architectural design is described, followed by the implementation details, the performance test results, and the user trials we carried out. We conclude by discussing the limitations of the current system, where future R&D is still needed, and what some of the possible future uses might be.

1. Introduction

Many organisations would like to develop stronger bonds with their customers and stakeholders, and give them personalised access to their web services. Universities are not an exception to this. Lifelong learning, student retention and alumni relations are all drivers towards strengthening the bonds between universities, their staff, and current and future students. The Logins4Life project was devised as an enabler of this, by allowing current and future staff and students to have account login access to university resources from the time of their first interaction with the university, until potentially the grave. However, the university did not want to proliferate the number of usernames and passwords that users need to remember. Instead it wanted to let users leverage their existing accounts, at social network and other web sites, to access university resources.

The university launched the Logins4Life project in order to federate existing external accounts, such as Facebook, Google etc. with its internal accounts, thereby allowing users to use their existing accounts to access university resources. This presents a number of challenges, since the university does not know which accounts a user might already have, and it might not have been in contact with the user prior to their first attempt to access university resources. Consequently user self-management of accounts must be adopted. However different external sites will have different authentication procedures and technologies, and therefore the level of assurance (LoA) [8] that each external site provides will be different. If the university resources are to remain secure, the LoA of the authenticating site must be taken into account when granting access. Finally the whole process must be very easy to use otherwise users will soon get frustrated and lose interest.

The rest of this paper is structured as follows. Section 2 reviews related R&D. Section 3 describes the design of the system. Section 4 describes the implementation. Section 5 presents the performance and load testing results, and Section 6 the user trials. Section 7 concludes with a discussion of the implementation, presenting the limitations of the system and the subsequent enhancements to fix

these, and our future plans. The innovative features of this project are as follows: it allows (unknown) users to utilise university resources and obtain a personalised service, without having to first register for an account at the university; it demonstrates effective interoperability between different open and proprietary authentication mechanisms and protocols whilst pointing out their current limitations; it shows how the LoA concept can be effectively used to provide coarse grained access controls; it leverages existing standards (and open source implementations) and adds enhancements to these to rectify their current deficiencies, and it provides a user friendly account management system which allows users to choose which of their existing IdP accounts they wish to link together to gain increased access to the university's resources.

2. Related Research

Authentication is a key functionality of federated identity management (FIM) systems, and it has three components: the protocol that is used to communicate the authentication assertion between the identity provider (IdP) and service provider (SP), the authentication mechanism and tokens that are used to authenticate the user by the IdP, and the authentication procedure that is used by the IdP during user enrolment and registration.

FIM systems are now well established and a variety of open protocols are in use between the IdP and SP. The UK Access Management Federation (UK-AMF), which is based on the Shibboleth protocol and implementation [2], has 845 member organisations [4] which include over 220 SPs. The Shibboleth protocol is currently based on SAMLv2 [3] and the Shibboleth team have contributed significantly to its development (and are continuing to lead enhancements to SAML within the OASIS Security Services Technical Committee). Another major protocol development is that of OpenID [5], originally started by Six Apart but it soon involved major players such as Microsoft, AOL, Google and Symantec. OpenID now boasts that 50,000 web sites can be accessed using this protocol [6]. Microsoft also developed its own open protocol based FIM system called CardSpace [7], but this was never a success, unlike its predecessor, the proprietary Passport system, which did gain some limited take up. Microsoft has now combined OpenID, Passport and CardSpace into its Windows Live ID single sign on protocol suite, but this is primarily used for accessing Microsoft sites only such as Hotmail, MSN and X-box live.

Unfortunately not all major web sites support open protocols, with sites like Facebook and Twitter using their own proprietary mechanisms. Consequently there is no universal single sign on protocol for FIM systems. Any solution that is devised will need to address this aspect and be extensible in order to cater for future single sign on (SSO) protocols.

The other two components of authentication i.e. the user authentication mechanism and tokens, and the enrolment and registration procedure, are addressed by NIST in its special publication "Electronic Authentication Guideline" [8]. This defines the Level of Assurance (LoA) concept which combines all three components together to create a metric in the range from 1 to 4. 1 is the lowest/weakest level of assurance and does not require the user to have been physically identified during registration, but it does assert that it is the same online user each time (regardless of who this user actually is). LoAs 1 and 2 allow password based authentication mechanisms. 4 is the strongest/highest level of assurance and requires that the user's identity has been verified physically during registration, using official documents such as a passport and birth certificate, and that online authentication is carried out using strong cryptography where the user's key is held in a tamperproof hardware device. Any FIM system that supports multiple authentication protocols and mechanisms must include the LoA concept in the design if the final system is to be secure.

Access control, or user authorisation, is another important functionality of FIM systems. This controls which resources at the service provider's site the user is authorised to access. Attribute based access control (ABAC) is a much more scalable solution than identifier based access control

lists. ABAC was first standardised over 15 years ago in ISO 10181-3 [9], although the term ABAC was not coined until more recently. The ISO model comprises an application dependent policy enforcement point (PEP) – termed the access control enforcement function in [9] – which receives the protocol messages from the IdP, makes a call to the application independent policy decision point (PDP) – termed the access control decision function in [9] – and based on the decision either grants or denies the user access to its resource. Later enhancements to the ISO model by the OASIS XACML TC [10] have introduced a policy administration point (PAP) for storing policies, a policy information point (PIP) for retrieving additional attributes needed by the PDP to make a decision, and a context handler that orchestrates the protocol exchanges between the various components. The XACML TC has also published the SAML-XACML protocol [11] which allows remote PEPs and PDPs to communicate securely with each other. Various policy languages have been defined for the PDP over the last decade, including XACML [10], X-Sec [12], PERMIS [13], SecPAL [14], and many others. New ones are still being developed e.g. PrimePrivacy [15]. Consequently any FIM system should be independent of the PDP policy language. The SAML-XACML protocol facilitates this.

Proxy IdPs are another useful component in identity management systems. A proxy IdP appears to be a normal IdP to an SP, whilst it is in fact a gateway to other (hidden) IdPs. Proxy IdPs have been implemented in several projects prior to our own, with myVOCS [21] being one of the earlier examples. The weakness of the proxyIdP model in general is that the SP has to trust the assertions made by the proxyIdP, without knowing if the proxyIdP is the true source of the assertions or not. We solve this trust dilemma in our system by incorporating the proxyIdP in the SPs’ trust domain so that the SPs know they can trust the assertions it makes.

3. Architectural Design

3.1. Authenticating the User

The overall architecture is shown in Figure 1. At the centre of the system is a proxy IdP which is run by the university and is therefore implicitly trusted by the university’s service providers (SPs).

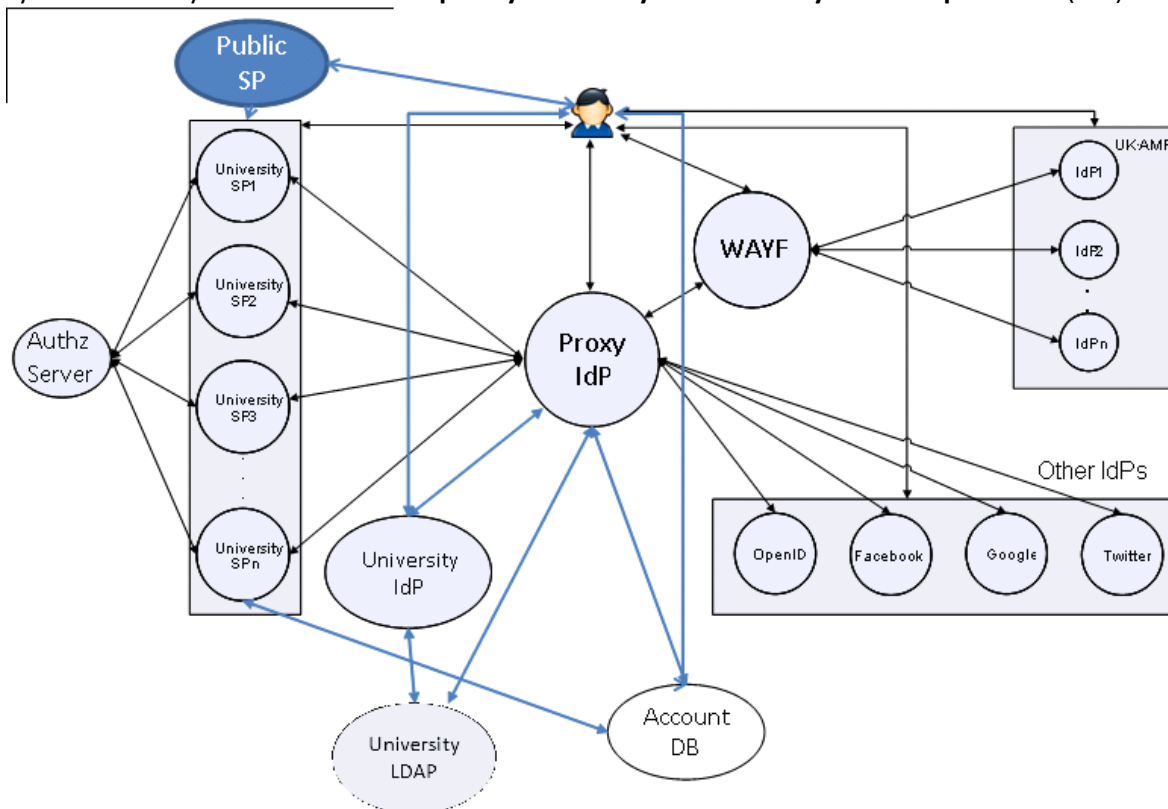


Figure 1. Overall Architecture

A user may access public services (represented by Public SP in Figure 1) without any restrictions. However, whenever the user attempts to access any protected service at the university (designated SP1....SPn in Figure 1), the user is redirected to the proxy IdP and is invited to login. The SP indicates the minimum level of assurance (LoA) it requires from an authenticating IdP and this cause the proxy IdP to tailor the login screen to only include those IdPs which it knows can equal or surpass the requested LoA. The actual login screen of the current proxy IdP implementation is shown in Figures 2 and 3 for different LoAs.

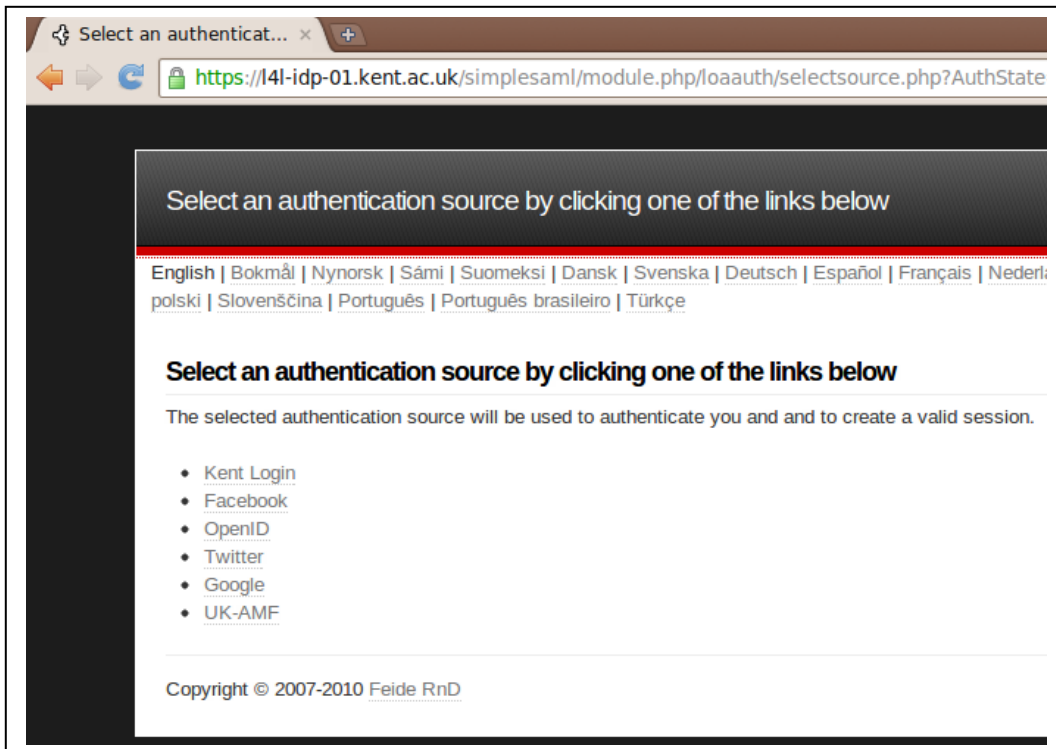


Figure 2. The Proxy IdP's Login Screen for LoA values ≥ 1 .

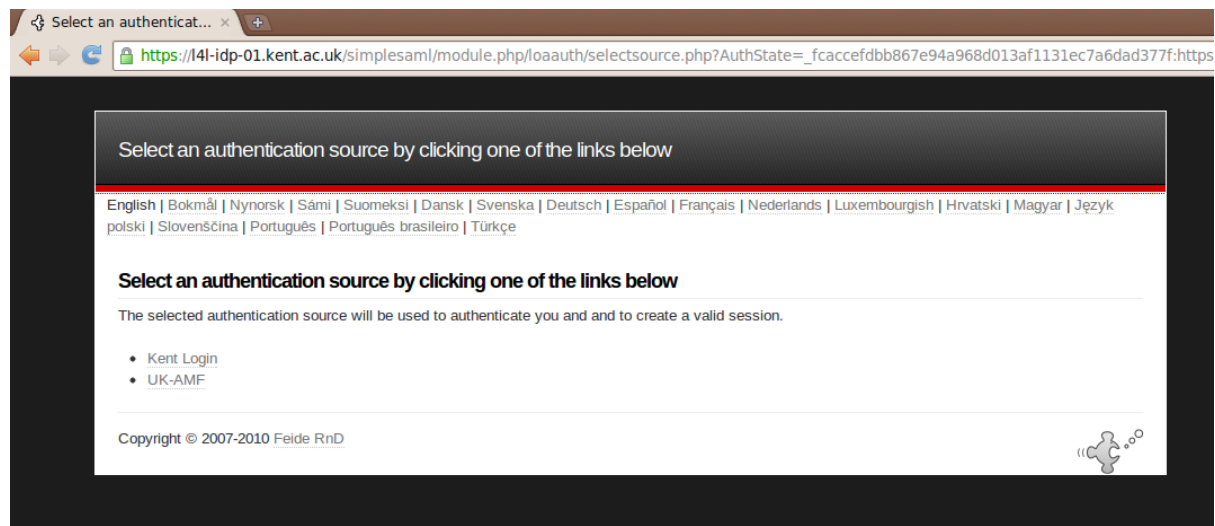


Figure 3: Login screen for LoA value values ≥ 2 .

The user chooses his preferred IdP and is redirected there by the proxy IdP. In the case where the user chooses the UK Access Management Federation (UK-AMF) as his IdP, the user is redirected to

the UK-AMF's Where Are You From (WAYF) service, which allows the user to choose which UK university he is from before being redirected there to login.

The proxy IdP is capable of communicating with each IdP in its supported protocol, and this shields the University SPs from needing to know these different protocols. It also allows new protocols to be seamlessly added in future. The SPs talk to the proxy IdP using a single standard protocol (currently SAMLv2.0).

The user's chosen IdP authenticates the user in its own proprietary way (which is almost invariably by username and password). In the case of the University IdP, it utilises the University's LDAP service to perform authentication, since the latter holds the usernames and passwords of all university registered users. The chosen IdP returns an authentication assertion to the proxy IdP, in its particular protocol. The authentication assertion contains either the actual username of the user at the IdP e.g. an OpenID, or a persistent identifier (Pid) which privacy protects the name of the user e.g. from a UK-AMF IdP. The trusted UK-AMF IdPs are also asked to provide attributes of the user which will subsequently be used by the PDP for fine grained access controls (see later).

The proxy IdP validates the assertions that it has received, and keeps their content, discarding their signatures and other protocol wrappings.

3.2. Utilising the LoA

The proxy IdP is configured with the LoAs of each of the configured protocol types, using the NIST guidelines. In the case of the SAMLv2 protocol via the UK-AMF IdP's, whose trusted meta data is distributed to the proxy IdP, these are given an LoA of 2. The same value is given to the University's own IdP. All the other IdPs (OpenID, Facebook etc.) are assigned an LoA of 1. Whilst each IdP might be using the same authentication mechanism (typically username and password) only the UK-AMF IdPs (including the University's IdP) have performed face to face authentication at registration time and know the true identity of the user. The other IdP's typically do not perform any user authentication at registration time (other than perhaps verifying the email address), and therefore the user could be anyone. This lack of assurance is reflected in the lower LoA value. A future enhancement could be to allow trusted IdPs, which are able to authenticate users by a variety of means, to dynamically specify the actual LoA they used for each authentication session, rather than having a configured static value as now.

If a user has linked an external account with an LoA of 1 to an internal account with an LoA of 2 (see below) and the user logs in via this external account, then the session is given an LoA value of 1.5. This recognises the increase in assurance which comes from the face to face registration of the internal account. The value of 1.5 is itself somewhat arbitrary, but we needed a way of recognising the difference between authentication performed via a low assurance unlinked external account, a low assurance linked external account, and an internal account (which has the highest assurance). So we chose 1.5 as the mid-point. This does however indicate the rigidity and lack of granularity in the current NIST specification.

Once the authentication LoA has been determined, the proxy IdP creates its own authentication assertion (in SAML), inserts the LoA into it, signs it, and returns it to the SP. An attribute assertion is also included (as described below), which is used for authorisation purposes.

3.3. Authorisation

Authorisation comprises two parts: firstly retrieving the attributes of the user, and secondly making an access decision based on these. The proxy IdP is responsible for retrieving the validated attributes of the user, and passing them to the SP in an attribute assertion which it signs. The SP is responsible for calling the backend authorisation server to obtain an access decision. The authorisation server

talks a standardised secure authorisation decision protocol with each of the SPs, and supports multiple PDPs and policies.

The proxy IdP gets the user’s attributes from one of two sources: either the University’s LDAP server, or a UK-AMF IdP. Both sources are trusted. The LDAP server holds the attributes of all user’s registered at the university, such as their course of study, whether they are staff, postgraduate or undergraduate etc. UK-AMF IdPs hold similar information, but typically present this as attributes conforming to the EduPerson schema [1]. Users who are not registered at either of these places will not have any attributes presented to the SP and therefore will gain access to fewer resources. Users who are registered at the university, but who might have authenticated via an external IdP such as Facebook, can gain additional authorisation privileges by linking their internal and external accounts together as described below. In this way the proxy IdP can still retrieve their attributes from the university’s LDAP server.

Each PDP in the authorisation server is configured with a hybrid RBAC-ABAC policy in which the LoA is modelled as a hierarchical role, in which level $4 > 3 > 2 > 1.5 > 1$. In this way, any resource which requires an LoA of 1 to be accessed, will be accessible regardless of the actual LoA assigned to the user. The PDP’s policy can be as fine grained as needed, and its decisions can be based on any of the user’s attributes stored in either the University’s LDAP server or the UK-AFM’s remote IdPs. In our initial pilot implementation we only use the eduPersonAffiliation attribute which states which university the person is affiliated with.

3.4. The Account Database

The account database is a relational database, which forms the permanent storage for the proxy IdP. It contains two tables: a **User Table** that contains details of each individual authentication account and their Internal Identifiers and a **DN Table** that maps LDAP distinguished names to Internal Identifiers. Each entry in the User Table contains enough information about an authentication account to uniquely identify it without necessarily compromising user privacy (depending upon the particular IdP’s policy). An example of a User Table is presented below:

Internal ID	Persistent ID	IdP Entity ID	Level of Assurance	Nickname
User1	persistentID1	http://idp1.com	2	Fred
User2	persistentID1	http://somedp.com	1	Some Account
User1	http://openid/name	http://openid.org	1	OpenID
User3	Persist3	Kent.ac.uk	2	Kent Account

Table 1. An Example User Table

The Internal ID is used to link multiple account entries together and specify that they belong to the same user. In the above example User1 has two entries and has therefore linked those accounts together. The PersistentID and IdP Entity ID tuple is used in combination to uniquely identify a user account. This tuple must be used together as persistent id’s can only be guaranteed to be unique in the issuing IdP’s own security domain. The Level of Assurance field specifies how secure the service is deemed to be by the proxy IdP. The Nickname field is a usability optimisation that allows users to specify a user friendly and easily recognisable nickname that is displayed by the Account Management SP’s interface (see Figure 6), rather than displaying the machine generated Persistent IDs that are often returned by IdPs. Figure 4 shows typical PIDs that are returned by IdPs, and the user can click on these and replace them with a user friendly nickname (Figure 6).

If a university account is linked to an authentication account then the distinguished name of the user in the university’s LDAP service will be stored in the DN table. We separate this table from the main

user table as not all authentication accounts can be mapped to an internal DN. An entry in this table allows the proxy IdP to determine whether the authenticating user is a known member of the university or an unknown external user. The DN stored in this entry is used by the proxy IdP to retrieve the authorisation attributes to be sent to the SPs. An example DN table for the above user table is provided below:

Internal ID	Account DN
User 3	Cn=Persist3, ou=staff, o=kent.ac.uk, c=gb

Table 2. An Example DN Table

3.5. Account Linking

Before a user is known to the system, she will not have any records in the account database, regardless of whether she exists in the University LDAP system or not. The first time a (unknown) user interacts with the system and chooses to authenticate, the proxy IdP will receive a PId (or username) from the remote IdP. The proxy IdP searches for this combination in the account database, and upon not finding it, creates a new entry for this user. The user may decide to always use this same account, say Facebook, for accessing the university SPs, in which case this is all that will ever be stored in the account database.

If the user decides to use a different IdP account for accessing the university, then a second, unrelated entry in the account database will be created for the same user. At this point in time the proxy IdP does not know that these two accounts belong to the same user.

Whenever a user chooses to authenticate via the University IdP, this causes the proxy IdP to lookup the user in the University LDAP and retrieve her distinguished name (DN), and store it in the account database. The DN is subsequently used by the proxy IdP to retrieve the user's attributes for inclusion in the attribute assertion that is returned to the university's SP, as mentioned above.

The user can choose to add new accounts, link and/or delete existing accounts whenever she chooses via the Account Management SP (SPn in Figure 1). The Account Management SP can be seen as being a user centric configuration tool for the proxy IdP rather than a separate SP in its own right. Access to the Account Management SP is in the same way as to all the other federated university services and requires the user to login via the Proxy IdP (with an LoA value ≥ 1). Once authenticated, the user is shown the account management screen which displays the user's logged in account details (see Figure 4). The user may choose to link any of her other accounts to the current one(s) by selecting the Add New Account button, in which case the user is logged out of the current IdP and the login screen in Figure 2 is shown again. The user chooses a new IdP, is redirected to it, authenticates and is then redirected back to the proxy IdP. If the returned PId-IdP tuple is unknown to the proxy IdP, it inserts a new entry into its database. If the PId-IdP entry is found, nothing is done in terms of account management. The user is then redirected back to the account management SP for account linking to take place.

Account Management Service

Back to homepage	Provider	Account Name	LoA	Delete
Add New Account	www.facebook.com	100001460241965	1	Remove Account
	www.myopenid.com	http://ripulbd.myopenid.com/	1	Remove Account
	https://persistence.kent.ac.uk/idp/shibboleth	IR8+2dRYbfRtCVZ7UiCJVU5YCoA=	2	Remove Account

Figure 4. The Account Linking Screen

At the Account Management SP, if the account is not linked with any other accounts then it is simply added to the previous account of the user (as in Figure 4), but if the account is already linked with other accounts the user is asked if she wants to merge this account (and all of its existing linked accounts) with the previous account(s), or move the new account on its own, or cancel the operation without linking it to the previous one(s) (see Figure 5). In this way the user is left completely in charge of managing her identity, and can have as many sets of unlinked accounts as she wishes.

Account Linking Service

WARNING: the new account that you are trying to add to your set of linked accounts is already in a different set of linked accounts. Do you want to:

- Merge your two sets of linked accounts together?
- Move this account from its existing set to the current set?
- Leave the account where it is and cancel the linking?

[Submit](#)

Figure 5: Account Linking Page

At the account management page, the user can click the entry in the Account Name column to insert a user friendly nickname for each account, as illustrated in Figure 6.

Account Management Service

Provider	Account Name	LoA	Delete
kent.ac.uk	msf20	2	Remove Account
www.facebook.com	<input type="text" value="100001460241965"/> <input type="button" value="Save"/> <input type="button" value="Cancel"/>	1.5	Remove Account
www.myopenid.com	http://ripulbd.myopenid.com/	1.5	Remove Account
https://persistence.kent.ac.uk/idp/shibboleth	IR8+2dRYbfRtCVZ7UiCJVU5YCoA=	2	Remove Account

Figure 6: Setting a user friendly nickname for an account

The Account Management SP allows the user to link/unlink different accounts with each other and thereby increase/decrease the LoA values associated with her different accounts. In Figure 6 one can see that the Facebook and OpenID accounts have risen from LoA 1 in Figure 4 to 1.5 in Figure 6 through the linking of them with user's University of Kent account.

3.6. Conceptual Protocol Flow

A generalized conceptual flow for accessing protected pages at the SPs is presented below:

- I. A user tries to access a page protected by the SP for the first time, and sends an access request to the SP.
- II. The SP creates a SAML authentication request and sends it to the Proxy IdP.
- III. The Proxy IdP checks if there is a session for the user signifying the user is already authenticated. If there is extract the session parameters from its cache and goto XII.
- IV. Assuming there is no session, the Proxy IdP will present to the user all the available login options.
- V. The user selects one of the options. Let's assume the user chooses the option ABC.
- VI. The user is forwarded to the ABC site where the user authenticates herself.
- VII. Assuming a successful authentication results, the login information is returned to the Proxy IdP. This always contains the user's login PId at the authenticating site ABC. As there are different protocol modules for talking to the different authenticating sites, each module can also insert the following information into the reply, to be used later:
 - a. The type of the Authenticating site (Facebook, OpenID, Kent, UK-AMF etc.)
 - b. The LOA associated with the site/type
 - c. If the site is a UK-AMF site, the attributes returned from that site.
- VIII. The Proxy IdP checks if there is an entry for the pair (login PId, ABC) in the account database. If there is an entry, check if there is an associated DN with it. If there is not a DN goto XI. If there is a DN, go to X. If there is no entry in the account database then create one. If the type of the Authenticating site is not the local site then goto XI.
- IX. Retrieve the user's DN from the university LDAP, and add this DN into the account database.
- X. The user has an account in the university's LDAP. Read the attributes from LDAP using the DN.
- XI. The Proxy IdP creates a session and stores all the session parameters in its cache.
- XII. Create a SAML authentication and attribute assertion using the attributes retrieved from the university's LDAP and/or UK-AMF IdP (if any) and the LoA and send it to the SP.
- XIII. The SP extracts the required attributes and LoA from the assertion and creates an XACML request context using those attributes, the LoA and the user's request and sends it to the authorisation server.

- XIV. The authorisation server's PDP analyzes the attributes and LoA and decides to either grant or deny access to the resource based on the policy, and sends the decision back to the SP. If the request is granted, the user will be forwarded to the requested page.

4. Implementation

3.1. Protocols

The login process is invoked when a user attempts to access a protected resource at a university SP. The SP sends a SAMLv2.0 authentication request to the proxy IdP, formatted according to the Web Browser SSO Profile (defined in Section 4.1 of [16]). This message has been tailored by setting the <authnContextClassRef> element of the <RequestedAuthnContext> to the URN which identifies the specific LoA that is required to access the service. Thus each SP is configured with the minimum LoA required to gain access.

The current SAMLv2.0 specification is deficient in that it is not possible to dynamically request a set of attributes at authentication time (although it is possible if two round trips are performed, the first to authenticate the user only, and the second to request the user's attributes only). All the SP can currently do is set the <AttributeConsumingServiceIndex> attribute in the authentication request to a predefined integer value which the IdP is meant to map into an already known set of attributes. We have extended the SAMLv2.0 specification by defining a new request type: the <AuthnAttributeRequest> message, which allows the SP to dynamically request the set of attributes it requires along with its current authentication request [17]. Whilst we can use this new message type between the university SPs and the proxy IdP, we cannot use it with existing SAMLv2.0 IdPs as they will not understand it. Therefore in the current implementation the proxy IdP sets the <AttributeConsumingServiceIndex> attribute to the value 0, which means ALL.

When the proxy IdP receives the SAML v2.0 authentication request it determines which of its configured IdPs produce a high enough LoA value to be equal to or greater than the value specified in the <authnContextClassRef> element. It then displays these choices to the user (see Figure2) who chooses one of them via a HTTP exchange. The proxy IdP is configured with a set of modules that talk the various supported protocols. It utilises the appropriate protocol module to access the chosen IdP, which performs user authentication and returns an authentication assertion to the module. In the case where the chosen IdP is a UK-AMF IdP, the module formulates a standard SAMLv2 authentication request and asks the IdP to return all the user's attributes as well, as specified above.

The protocol module validates the received assertion, extracts the PId/IdP tuple (and attributes for the UK-AMF module), appends the configured LoA for this protocol/IdP type, then returns these to the proxy IdP. The proxy IdP accesses its database and attempts to find a match in the user account table. If the account is found the proxy IdP also determines if an internal LDAP DN exists, and if so, it pulls additional attributes from the university's LDAP server according to its preconfigured Attribute Release Policy (ARP). The proxy IdP creates a SAMLv 2.0 <samlp:Response> message for the requesting SP, which contains an authentication and attribute assertion. The assertion is signed and encrypted for the SP. The proxy IdP is the trusted issuer of the assertion.

When the SP receives the <samlp:Response> message it validates the assertion. This involves decrypting the SSO assertion using its private key and verifying the signature with the public key of the proxy IdP to ensure message validity and confidentiality. Finally it checks that it is indeed the intended recipient of the assertion, using the <AudienceRestriction> element, and that the assertion is still valid timewise. Each user attribute contained in the SSO assertion (including the LoA) is parsed and mapped into its equivalent XACML counterpart and added to an XACML request context.

The SP now talks to the backend authorisation server using the SAML-XACML protocol [11]. This request contains a standard XACML request-response context wrapped as a new type of SAML assertion – the XACMLAuthzDecisionQuery assertion. By utilising the SAML-XACML protocol we can add additional levels of security to the request, such as authentication and encryption. In our implementation we use TLS. We can also specify the authorisation policy to be used by the authorisation server to process the SP's request. This allows us to support use cases where the authorisation server contains multiple PDPs and multiple policies, and could have a different policy for each SP. The protocol allows either a Policy Reference or a full policy to be placed in the body of the XACMLAuthzDecisionQuery request, and this policy should be used to authorise the user's request.

The current version of simpleSAMLphp which is used in our implementation does not provide a mechanism for the SP to extract the <AuthnContextClassRef> element from the response message, in order to retrieve the actual LoA value of the session. We had two implementation options in order to provide LoA based authorisation:

- i) we could change the simpleSAMLphp source code to extract the field. However this approach potentially produces a long term support overhead each time the next release of simpleSAMLphp is released (unless our code is adopted by the distributors). Furthermore the SP still has to pass the LoA to the authorisation server in some way, so it was deemed to be unsuitable;
- ii) we could tailor the proxy IdP to add the LoA value to the set of subject attributes in the attribute assertion of the SAMLv2.0 response it creates and sends to the SP. The SP then transparently copies this attribute over into the XACML request context, along with all the other user attributes. The PDP policy then treats the LoA as just another user attribute when making an authorisation decision. This is the method we adopted.

When the authorisation server receives the SAML-XACML request it determines which of its PDPs to use to evaluate the request. Once a PDP(s) has been chosen this evaluates the XACML request and returns an XACML response. The authorisation server responds to the SP with an SAML-XACML response message. If a DENY response is returned the user is redirected to an authorisation denied page that provides the user with links to logout of her current SSO session and re-authenticate using a different account. If a GRANT response is returned the requested page is displayed.

4.2 Proxy IdP

The proxy IdP software has been implemented using SimpleSAMLphp [20]. This supports module-based design and has different modules for implementing different authentication mechanisms, e.g. OpenID, Facebook, Twitter, etc., with the ability to add new modules as desired. The Twitter, OpenID and Google modules worked out of the box, but the AuthFacebook module had to be re-written as the provided one did not work correctly (it was based on an old Facebook API specification).

We created two new modules for our proxy IdP. The first, loauth, allows the set of IdPs to be filtered according to the LoA requested by the SP in the <authnContextClassRef> inside the SAML authentication request. This ensures that IdPs whose LoAs are known to be lower than that requested by the SP are filtered out so that only IdPs whose LoA are sufficient will be displayed to the user during the authentication phase.

simpleSAMLphp has the concept of an authentication processing filter that allows additional tasks to be performed after the authentication is complete and the user has been re-directed back to the proxy IdP. We created one new authentication processing filter, the ldapcheck module, which implements steps VII-X of the protocol flow described in Section 3.6 i.e. checks if the user has an

entry in the account database, and if not creates one. It also checks if the user has an LDAP entry and if so picks up the user's attributes from the university's LDAP server.

4.3 Account Management SP

The database that stores the relations between accounts is implemented as an Axis2 service. This service's sole purpose is to provide a web interface to a backend database that contains user linking data. As we wished to provide a generic service for all possible database types all invoking calls are passed to an instance of the database interface and the results formatted and returned to the requestor. The database interface itself provides multiple account management specific methods for storing, editing and deleting entries. This meant that the PHP frontend can call the service by parsing the service's WSDL and can receive properly formatted responses no matter what backend database is currently instantiated by the service. The default instantiation uses Apache Derby Java libraries to embed a SQL based relational database inside the service providing persistent but configuration free account storage. The backend database however can easily be changed at run time for a user supplied database instance by editing the service's Spring configuration file prior to launch.

4.4 The Authorisation Server

The authorisation server is the advanced authorisation service as described in [18]. This supports multiple PDPs, multiple policy languages, obligation handling, Break the Glass policies and several other advanced features.

3. Performance and Load Testing

Since the proxy IdP is core to the entire system, we needed to run performance and load tests to see how many users it could concurrently support, as well as to find the CPU load, memory usage and time taken for different numbers of users logging in concurrently. Originally we wanted to determine the maximum number of users the proxyIdP could support concurrently, and how long this maximum could be sustained before the system failed. However, the load testing program we were using (JMeter) was unable to support the required number of threads to allow this, and reached its limit at just over 300 threads/users, well before the cpu or memory were exhausted. Due to this limitation we could only test 1, 2, 5, 10, 20, 50, 100, 200 and 300 concurrent users. Each test was run five times to increase the accuracy of the collected data. When a given user had completed the test once, it immediately tried again, until it had completed the test 5 times. The computer used to run the proxyIdP tests had the following specification: Processor - 2x Intel(R) Core(TM)2 Duo, CPU E8400 @ 3.00GHz, Memory - 3317MB, OS - Ubuntu 9.10. The software used was Apache JMeter version 2.4.

We ran four sets of tests. In the first set, the users tried to access the account management page, were redirected to the proxy IdP to login, selected the university IdP, logged in via this and then were presented with the required page. In the second set, the users selected Twitter to login instead of the university IdP. In the third set, the users tried to access a university SP which required an LoA of 1, chose Twitter to login, and were presented with the required page. In the final tests, the users tried to access a SP which required an LoA of 1.5, chose Twitter, were denied access and shown an error page informing them that they were unauthorised and giving them options to cancel or log in with a higher LOA.

The results are presented below. Figure 7 shows the time taken for a given number of users to access the protected page 5 times, which ranged from 4 seconds for a single login via the university IdP to 870 seconds for 300 users concurrently logging in via Twitter.

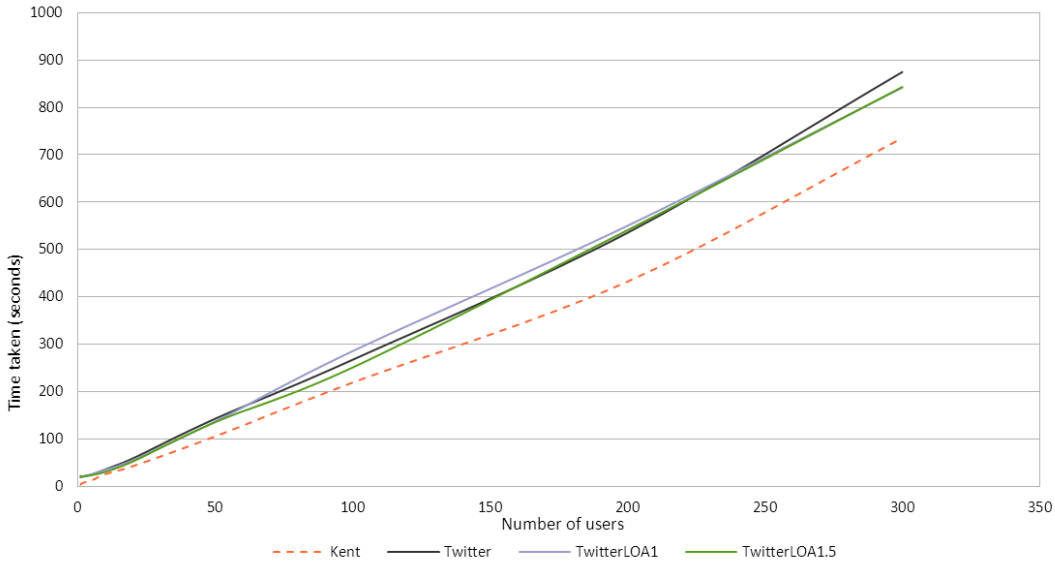


Figure 7. Time taken for concurrent logins

Figure 8 shows the CPU load, which quickly rose to over 50% and then levelled out when 50 or more users tried to log in concurrently. We assume this levelling is due to a bottleneck in either the remote IdP or the network or both, but is not due to the proxy IdP.

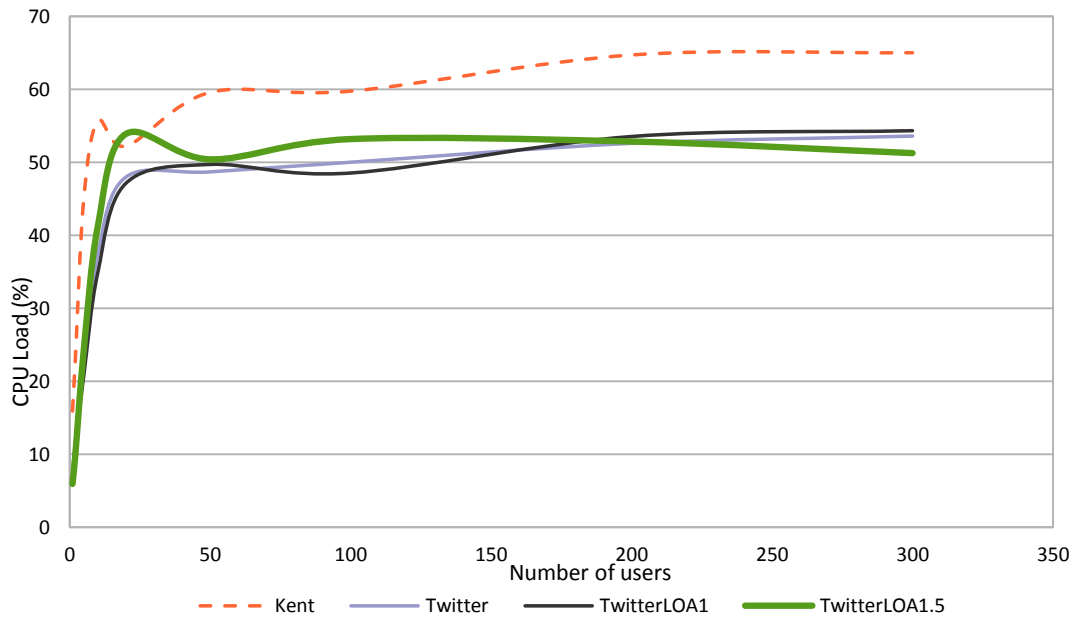


Figure 8. CPU load for concurrent logins

Figure 9 shows the memory usage, which steadily rises as the number of users increase, and then levels out at just over 800 Mbytes, which is only 25% of that available. This shows that the bottleneck is not with the proxy IdP but is with external resources.

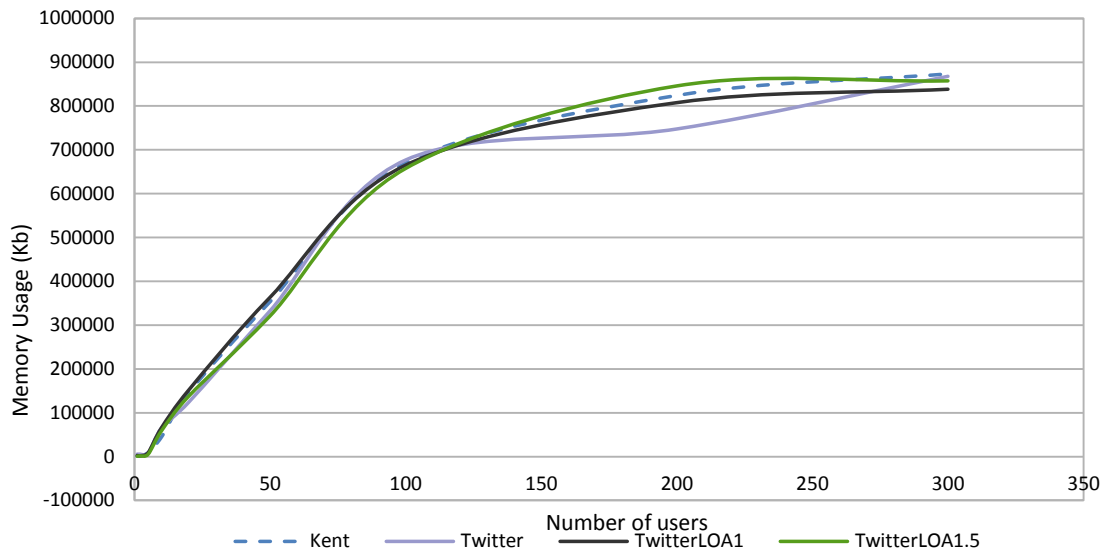


Figure 9. Memory usage for concurrent logins.

4. User Trials

The user trials were designed to determine the ease of use of the system, and the level of understanding an external, novice user would achieve from using the system for the first time. Users were asked to perform the following four tasks, and to “think aloud” whilst doing so. This allowed the observer to record the user’s thought processes for later analysis so as to discover where any usability problems might lie.

4.1. User Tasks

1. Download the University of Kent postgraduate application form.

This task required an LoA of 1 which could be achieved by logging in with any IdP account, navigating to the postgraduate studies section, and then selecting a hyper link to download the .doc form. (Subsequent enhances to the application will allow the user to upload the completed application form and track progress of his application, by using the same IdP account to login.)

2. Determine which lecture is in week 6 for module CO876.

This task requires the user to have an LoA of at least 1.5 and an eduPersonAffiliation of “University of Kent”. This LoA is achievable by either logging in via the university IdP or a UK-AMF IdP, or with any other IdP which has previously been linked to a university account. However only the first and last options provide the correct affiliation. This task involves navigating to the Moodle section and then following the link to the CO876 page of Moodle. Users can then scroll down the page to week 6 or alternately download the timetable from a link on the same page.

3. Download the paper “An Advanced Policy Based Authorisation Infrastructure” from the ACM digital library.

This task requires an LoA of 2 which can be achieved by either logging in via the university IdP or a UK-AMF IdP. It involves navigating to the ACM digital library section using a menu button, and then selecting a hyper link to the .pdf paper.

4. Access the Student Data System (SDS)

This task requires an LoA of 2 and additionally an eduPersonAffiliation of “University of Kent”. This combination is only achieved by logging in via the university IdP.

Each user was given specific instructions as to which accounts they should use in order to complete the tasks. All users who underwent the trials already had their own Facebook and university accounts (at least).

1. User 1 was asked to use any of their existing accounts as they saw fit.
2. User 2 was asked to use only their Facebook account.
3. User 3 was asked to use only their university account.
4. User 4 was asked to login to their Facebook page prior to the start of the test and then subsequently to use any account of their choosing to complete the tasks.
5. User 5 was given a UK-AMF account un/pw and asked to use this account first and then any other account of their choosing.
6. User 6 was the same as User 5, but after performing the 4 tasks was specifically asked to link his accounts together and then repeat the 4 tasks again.

4.2. User Trial Results

The tests were designed so that all users, except User 2, would be able to complete all four tasks. User 2 would only be able to complete Task 1. The results that were obtained were as expected. However, the end results on their own tell us little about the ease of use of the system. For this, we need to analyse the tape recordings of the users "thinking aloud". In some cases the users proceeded without difficulty and were able to complete the tasks in a very straightforward manner requiring little comment. However, there was one incident in which the user became confused as to what was happening, and as to why they could not access certain pages.

A breakdown of each user experience is given below.

1. User 1 - used their university account to login for task 1 and as a result was authenticated to LoA 2 and had an affiliation of University of Kent, which were sufficient for all the following tasks. Due to this, all tasks were completed without any issues. After completing the tasks User 1 voluntarily linked his Twitter, Google, and Facebook accounts to his Kent account, and he also logged out of the system. He then logged back in with his Facebook account which he used to access Moodle.
2. User 2 - was asked to login using their Facebook account and was therefore unable to complete any tasks other than task 1. Despite this the user noted that they understood why access was being denied as they had only used Facebook to log in.
3. User 3 - was asked to use their university login account and because of this each task was carried out without issue. The user mentioned that everything was easy to find and access.
4. User 4 - was asked to login to their own Facebook page prior to starting the test which they did. We anticipated this would give them SSO for task 1. However, when starting task 1, the user was presented with the list of log in options (Figure 2) and they chose to use Google. Consequently they had to log in again using the Google authentication system. Had they chosen the Facebook option they would not have had to enter any log in details again. When the user attempted to access Moodle to complete task 2 they were informed that they needed a higher level of authentication to access the section. This told them that they should log in with a different account, and the university and UK-AMF options were displayed (Figure 4). The user then chose to log in with their university account which allowed them to proceed with all of the remaining tasks without further issue. No account linking was attempted. After the test, they were asked if they found it easy to determine which account they should use in order to access the required resources and they answered that they thought it was fairly obvious!
5. User 5 - was given account details for a UK-AMF account and asked to use this initially, followed by any other accounts as he saw fit. For task 1 he logged in using the UK-AMF option and completed the task without issue. When he attempted task 2 he was told he was not authorised to access the Moodle service. He mentioned aloud at this point that he must need to login with his university account as the University's Moodle service is only for local

students. He returned to the home page, logged out, reattempted to access Moodle, was prompted to login, chose his University account and was granted access to Moodle. All subsequent tasks were completed successfully. He did not choose to access the Account Management service.

6. User 6 - was given the same instructions as User 5 but after he had completed the 4 tasks he was asked to access the Account Management service and link some of his accounts together, then log out and repeat tasks 1-4 again. The results were interesting. Task 1 was completed as per User 5, but when User 6 attempted task 2 and was prompted to login again he chose the Facebook option, logged into his Facebook account and subsequently was told that the account did not have enough access rights. This pointed to a user interface design issue which is discussed below. User 6 then chose his university account and successfully accessed Moodle and completed the remaining 2 tasks. The user then accessed the Account Management section and linked his Facebook account to his university account (which he was already logged in with). He noted aloud that the name returned by Facebook was just a collection of random numbers, so he clicked on the name and saw he could rename it, so he renamed it to "Alex's Facebook". He then linked his Twitter account, and mentioned that as the displayed name was the same as his username for Twitter it was fine so he didn't need to rename it. Finally he linked his Google account, the default name of which is also a long string of numbers, he noted this and renamed it to "My gmail". After this he returned to the homepage and logged out, before attempting task 1 again. Task 1 was completed successfully using the UK-AMF account he had been given. When he attempted to complete task 2 he was again told he was not authorised to access the resource. He returned to the home page, logged out, and then clicked the Moodle link again. He again tried to use his Facebook account to login (at this point he also noticed that he was already logged into Facebook as it supports SSO and so didn't have to re-enter his details) and this time he was granted access to Moodle. He said aloud that this was probably because he had linked it to his university account (which is correct). Task 3 proceeded without issue. However, when he attempted to access SDS he was told he needed a higher level of authentication. At first he tried to use his new UK-AMF account but was told he didn't have access. This is a symptom of the attribute selection problem that is discussed below. He said aloud that this was because the SDS was only accessible to university students (which is correct) and so he logged out and then reattempted the task with his university account, which allowed him to successfully complete the task. It should be noted that if User 6 had linked his new UK-AMF account to his university account then he would have been granted access to SDS, but one surmises that he did not regard this as being "his" account and therefore did not link it, as he did successfully link all his other accounts (that rightfully were his).

In general there was not a great deal of "thinking aloud" during the tests as most of the tests were quite straightforward. Users mostly mentioned that they found the tasks easy to complete, and that the system was simple to understand. Two users (1 and 6) noted that it would have been helpful to know which account they were currently logged into, and that perhaps the pages should display a message such as "Logged in via Facebook". This has now been added. Another user mentioned that it helped them to know which account to use to login to more sensitive pages because only the applicable log in options were displayed. User 3 commented that the page layout was good and easy to navigate.

5. Discussion and Conclusion

The project has achieved its initial objectives and allows external users to easily access university resources using their existing external login accounts, without having to first register at the university for a new account (which job and postgraduate applicants currently have to do). Furthermore, existing students who have both university and external accounts such as Facebook,

can choose to link these together, and afterwards, will be able to seamlessly access restricted university resources, such as Moodle, using the SSO features of Facebook. Since many students tend to login to Facebook as one of the first tasks they do every day (before attending lectures!) then they will no longer need to login to their university accounts in order to access lecture material on Moodle.

One user interface design issue was uncovered by User 6. Facebook (and other external accounts) have an LoA of both 1 and 1.5 depending on whether the account is already linked with a university account or not. But before user authentication is performed, there is no way for the proxy IdP to know if this user has linked his accounts or not. So although Moodle says it requires an LoA of 1.5 the GUI displayed all of the LoA 1 accounts as well as the other higher LoA accounts. This could result in the user choosing an account and still being refused access. The solution was to redesign the Login screen to have two sections. The top section now displays accounts known to be always at or above the requested LoA and says the user can choose any of these IDPs to login, whereas the bottom section displays accounts that might be at or above the chosen LoA, and says the user can choose any of these accounts if she has already linked it to her university account.

An outstanding major issue is that of attribute selection (which User 6 encountered). There is currently no standard way in SAML of dynamically requesting a set of attributes in the Authentication Request, and even if there was, there is no way for the proxy IdP to know which external IdPs can provide which (subset) of these. We have extended the SAML protocol to enable the former [17] and enhanced the proxy IdP to request and store a list of attribute types with each account entry. But this requires extensions to the IdP protocols, and therefore is not feasible without significant standardisation effort. It also requires an enhanced GUI to filter out inappropriate IdPs and allow the user to select between the remaining ones based on the attributes and LoAs they can provide. In a related project we built such an interface based on enhanced CardSpace protocols and designs [19]. However in the current system we simply pick up all the user's attributes from the local LDAP and the UK-AMF IdPs and none from any of the other IdPs. This does cause user inconvenience sometimes, as was shown with User 6. It is still an active research topic how to provide an ideal solution to this problem.

One major issue that currently has not been addressed is how to retire dead accounts. We expect that the account database will grow to many hundreds of thousands of entries over time, and that some users will lose interest, move abroad, forget all their passwords or even die, before removing their old accounts from the system. Thus the university has to devise a way of removing dead accounts without removing infrequently used though still active accounts. Annually emailing each user is one proposed solution, but we recognise that this does not fully address the issue. The topic of Digital Death [22] is starting to attracting significant attention in the identity management world.

There are many as yet unexplored opportunities for leveraging the Logins4Life system. Password management is one of them. This is a costly exercise for many organisations. When students forget their university passwords today, they have to go through a manual procedure which involves turning up with their ID card at a help desk, and being issued with a new password. If their university account is linked to an external account, it should be possible, after logging in via the external account, to request a new university password be emailed to them. Providing the authentication mechanisms (typically un/pw) of the external account and the email system are as good as the university's mechanism, then the same level of assurance will have been obtained.

The software that we have developed is being released as open source code, both through the university's web site and Feide, the authors of simpleSAMLphp. A public demonstration (and participation in the trial tests) is available at <https://persistence.kent.ac.uk/logins4life/>.

Acknowledgements

The authors would like to thank the UK JISC for funding this research under their Access and Identity Management Programme.

References

- [1] EduPerson schema. Available from <http://middleware.internet2.edu/eduperson/> (last accessed 9 March 11)
- [2] R. L. "Bob" Morgan, Scott Cantor, Steven Carmody, Walter Hoehn, and Ken Klingenstein. "Federated Security: The Shibboleth Approach". *Educause Quarterly*. Volume 27, Number 4, 2004
- [3] OASIS. "Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0", OASIS Standard, 15 March 2005
- [4] See <http://www.ukfederation.org.uk/content/Documents/MemberList> (last accessed 12 Feb. 11)
- [5] "OpenID Authentication 2.0 – Final". Dec 5th 2007. Available from http://openid.net/specs/openid-authentication-2_0.html
- [6] See <https://openid.org/home> (last accessed 12 Feb. 11)
- [7] David Chappell. "Introducing Windows CardSpace". MSDN. April 2006. Available from <http://msdn.microsoft.com/en-us/library/aa480189.aspx>
- [8] William E. Burr, Donna F. Dodson, Ray A. Perlner, W. Timothy Polk, Sarbari Gupta, Emad A. Nabbus. "Electronic Authentication Guideline", NIST Special Publication 800-63-1, Feb 2008
- [9] ITU-T Rec X.812 (1995) | ISO/IEC 10181-3:1996 "Security Frameworks for open systems: Access control framework"
- [10] OASIS "eXtensible Access Control Markup Language (XACML) Version 2.0" OASIS Standard, 1 Feb 2005
- [11] OASIS "SAML 2.0 profile of XACML, Version 2.0". OASIS committee specification 01, 10 August 2010
- [12] Bertino, E., Castano, S., Ferrari, E. "On specifying security policies for web documents with an XML-based language". *Proceedings of the Sixth ACM Symposium on Access control models and technologies 2001*.
- [13] D.W.Chadwick, A. Otenko. "RBAC Policies in XML for X.509 Based Privilege Management" in *Security in the Information Society: Visions and Perspectives: IFIP TC11 17th Int. Conf. On Information Security (SEC2002)*, May 7-9, 2002, Cairo, Egypt. Ed. by M. A. Ghonaimy, M. T. El-Hadidi, H.K.Aslan, Kluwer Academic Publishers, pp 39-53.
- [14] Moritz Y. Becker, Cedric Fournet, and Andrew D. Gordon. 2010. "SecPAL: Design and semantics of a decentralized authorization language". *J. Comput. Secur.* 18, 4 (December 2010), 619-665.
- [15] Gregory Neven. "Privacy-enhanced access control in primelife". In *Proceedings of the 6th ACM workshop on Digital identity management (DIM '10)*. ACM, New York, NY, USA, 2010, pp 1-2. DOI=10.1145/1866855.1866857
- [16] OASIS "Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0", 15 March 2005
- [17] G.Inman, D.W.Chadwick. "SAML 2.0 SSO Extension for Dynamically Choosing Attribute Values"
- [18] D. W. Chadwick, K. Fatema. "An Advanced Policy Based Authorisation Infrastructure". *Proc DIM'09*, November 13, 2009, Chicago, Illinois, USA. ACM. pp81-84. ISBN:978-1-60558-786-8
- [19] David W Chadwick, George Inman, Paul Coxwell. "CardSpace in the Cloud". *Proceedings of the 17th ACM Conference on Computer and Communications Security*. ACM, New York, NY, USA. 2010, Pages: 657-659. doi 10.1145/1866307.1866388
- [20] SimpleSAMLphp is available from <http://simplesamlphp.org/>
- [21] Jill Gemmill, John-Paul Robinson, Tom Scavo, and Purushotham Bangalore. "Cross-domain authorization for federated virtual organizations using the myVocs collaboration environment". *Concurr. Comput. : Pract. Exper.* 21, 4 (March 2009), pp 509-532.
- [22] See <http://digitaldeathday.com/> (last accessed 9 March 2011)