Leveraging Social Networks to Gain Access to Organisational Resources

D.W.Chadwick
University of Kent
School of Computing
Canterbury
+44 1227 823 221
d.w.chadwick@kent.ac..uk

G.Inman
University of Kent
School of Computing
Canterbury
+44 1227 823 823
g.inman@kent.ac.uk

K.W.S.Siu
University of Kent
School of Computing
Canterbury
+44 1227 823 823
k.w.s.s@kent.ac.uk

M.S.Ferdous
University of Kent
School of Computing
Canterbury
+44 1227 823 823
ripul.b.d@gmail.com

ABSTRACT

We describe a federated identity management service that allows users to access organisational resources using their existing login accounts at social networking and other sites, without compromising the security of the organisation's resources. We utilise and extend the Level of Assurance (LoA) concept to ensure the organisation's site remains secure. Users are empowered to link together their various accounts, including their organizational one with an external one, so that the strongest registration procedure of one linked account can be leveraged by the other sites' login processes that have less stringent registration procedures. Coupled with attribute release from their organizational account, this allows users to escalate their privileges due to either an increased LoA, or additional attributes, or both. The conceptual and architectural designs are described, followed by the implementation details, the user trials we carried out, and a discussion of the current limitations of the system.

Categories and Subject Descriptors

D.4.6. Security and Protection, Access Controls

General Terms

Design, Security.

Keywords

federated identity management, level of assurance, social networks, authentication, authorisation.

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, past 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 unnecessarily proliferate the number of usernames and passwords that users need to remember. Instead it wanted to let users utilise their existing login accounts at social networking (SN) and other sites such as Google, to leverage their single sign on (SSO) capabilities when accessing university resources. A prospective student should be able to access low value resources at the university web site using their existing SN account, and, after registering as a student, be able to access more valuable resources, still using their SN account, but now benefitting from an increased assurance in his/her identity.

One major problem in using these SN and other sites for SSO, is that they perform little or no authentication of their users' identities at registration time. Thus there is no or very little binding between the virtual identity and the physical identity of a user. In terms of the NIST level of assurance (LoA) metric [8] this constitutes the lowest level. In comparison, the University of Kent's registration procedure is at a much higher level comprising face to face registration with a passport, which provides a very strong binding between the virtual and physical identities of a user. Any resource that should only be available to registered staff or students of the university cannot be made available to users who simply authenticate via a SN site, as they literally could be anyone. This is unfortunate, since the majority of Kent's students login to Facebook every day[1] but they cannot use its SSO facility to access campus resources. Instead they have to login again using their more strongly assured university provided username and password.

Another major problem is that some of these sites have very weak password policies, so it is relatively easy to masquerade as the site's user. Any system that leverages the SSO capabilities of these sites will need to be able to differentiate between sites that perform very weak login processes and those that do a much better job.

Leveraging SN and other sites thus presents a number of challenges. Firstly, the university does not know which SN accounts an existing or prospective student already has. Managing this would be costly to the university administration. User self-account management on the other hand would provide an acceptable low cost solution if it could be done securely. Secondly, the low assurance associated with these sites must be increased if their SSO mechanisms are to be adopted for access to valuable university resources, but not if the sites login process is

DIM'11, November , 2011, Chicago, Illinois, USA.

[1] In a student survey taken for this project, >80% of students used Facebook for more than 1 hour per day and >35% used it for >5 hours per day. In comparison about 30% used Kent's website > 1 hour per day and a negligible percent >5 hours per day.

unacceptably weak. Finally any system that is adopted must be very easy to use otherwise users will soon get frustrated and lose interest. We might summarise these challenges as "How can we leverage the ubiquity of social networking and similar sites to utilise their SSO mechanisms to access university resources, without compromising the security of the university's site".

The rest of this paper is structured as follows. Section 2 reviews related work. Section 3 describes the conceptual design of the system, and section 4 the architectural design. Section 5 describes the implementation. Section 6 presents the user trials. Section 7 concludes with a discussion of the implementation, its limitations, future enhancements and plans.

2. RELATED WORK

Authentication is a key functionality of federated identity management (FIM) systems, and according to the NIST special publication "Electronic Authentication Guideline" [8] 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 (here after called the login process), and the authentication procedure that is used by the IdP during user enrolment and registration (hereafter called the registration procedure). The Level of Assurance (LoA) concept combines all three components together to create a metric in the range from 1 to 4. All four levels require the assertion issued by the IdP to be cryptographically protected when transferred to the SP, but the registration procedure and login process vary from level to level.

LoA 1, defined as little or no confidence in the asserted identity's validity, is the lowest/weakest level of assurance and does not require the user to have been physically identified during registration. Remote anonymous registration is allowed. The login process allows medium strong passwords to be used provided they are not passed in the clear. The probability of correctly guessing a password during its lifetime needs to be worse than 1 in 1024 (or 10 bits of entropy), which can be achieved for example, by requiring users to choose 8 character passwords with a mixture of upper/lower case and special characters, allowing only 2 wrong password attempts per minute and requiring users to change their passwords every year. LoA 1 effectively assets that it is the same online user each time, but the SP can have little or no confidence in who this user actually is. Note that any IdP whose login process does not meet NIST's minimum requirements would not even qualify for LoA 1, and consequently should be assigned an LoA of 0. This effectively means that the SP cannot even have confidence that it is the same user each time.

LoA 2, defined as some confidence in the asserted user's identity, requires a government issued photo ID containing the user's address or nationality, but it allows remote registration with validation by a trusted registration authority. A pseudonym may be used for identifying the user instead of a meaningful name. The login process can still use passwords, but now the entropy must be greater than 14 bits (1 in 16,384), which can be obtained with a randomly generated 6 character password from a 94 character alphabet, or a user generated 16 character password, and the same false attempts profile as above.

LoA 3, defined as high confidence in the user's asserted identity, has the same registration requirements as LoA 2, but login must now be via proof of possession of a secret e.g. a digital signature or Kerberos exchange. A one-time password can be used providing there are more than 1 million values. Pseudonyms can no longer be used to identify the user.

LoA 4, defined as very high confidence in the user's asserted identity, is the highest level of assurance. It requires face to face registration with two official documents such as a passport and birth certificate. Online login must be carried out using strong cryptography where the user's key is held in a tamperproof hardware device.

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 uses digitally signed assertions. Another major protocol is OpenID [5] which boasts that 50,000 web sites can be accessed using it [6]. It uses symmetrically encrypted signatures to validate its assertions.

Not all major SNSs support open protocols however, with sites such as Facebook using their own proprietary protocols. However, Twitter uses Oauthv2 [7] and Facebook is currently migrating to Oauthv2.

When multiple protocols are involved in FIM, a proxy IdP is a useful component to act as a protocol gateway. It appears to be a normal IdP to the SP, talking its protocol, whilst it is in fact a gateway to other (hidden) IdPs which may talk different protocols. Proxy IdPs have been implemented in several FIM projects, with myVOCS [1] being one of the earlier examples. The weakness of the myVOCS model is that the SP has to trust the assertions made by the proxy IdP when it has no control over it and does not know the true source of the assertions that it receives. To overcome this, the proxy IdP needs to be part of the same trust domain as the SP.

Account linking, which we utilize in this paper, was introduced in our prior research on attribute aggregation [9]. We devised a trusted third party linking service that allows users to link their various SAMLv2 IdP accounts together. We further develop this service here, to be a multi-protocol proxy IdP gateway for an organisation.

3. CONCEPTUAL MODELS

3.1 LoA Model

Assuming the assertion protocol is always secure, then the NIST LoA concept comprises of two sets of variables: the registration procedure and the login process. The LoA of an IdP is determined by the lowest of these. If an IdP has a strong face to face registration procedure but a weak login process, then its overall LoA will be equal to that of the login process. Most SNSs have a very weak registration procedure so their LoAs are constrained to being ≤ 1, even if their login process is higher. Conversely many organisations have strong registration procedures requiring at least a government issued photo ID plus face to face registration, which could qualify as NIST LoA 4, but their login protocol might be much weaker. Conversely, Verisign class 1 certificates qualify for a login LoA of 3 but a registration LoA of 1. Our assertion is: if the strong registration procedure of an organisation can be securely combined with the strong login process of another IdP, whose registration LoA might be low, then the LoA of the user's SSO session between this IdP and the organisation's SP can potentially be raised to that of the organisation's registration procedure. We propose to achieve this as follows.

The first proposal is to give each IdP two LoA scores: a registration LoA and a login LoA. The registration LoA measures the strength of its strongest registration procedure, whilst the login LoA measures the strength of its strongest login process (noting that an IdP may have several different registration procedures and offer several different login methods). The IdP's LoA will then be the lowest of these two viz:

$$LoA_{IdP} = Min\ [RegLoA_{IdP}|LoginLoA_{IdP}]$$

The LoA of an IdP represents the most trust that a relying party can place in an IdP to authenticate its users, based on their registration and login methods. The relying party must obtain this information by some out of band mechanism e.g. in federation meta-data.

The second proposal is to introduce a session LoA. This is the LoA that the IdP assigns to a given user for any particular login session. The session LoA will always be lower or equal to the IdP's LoA. Factors that will make a session LoA lower than the IdP's LoA could be: the user chose to use a weaker authentication mechanism for this session ($LoginLoA_{UserIdP}$) or the LoA used a weaker procedure when registering this particular user ($RegLoA_{UserIdP}$), viz:

$$SessionLoA_{UserIdP} = Min\ [RegLoA_{UserIdP}|\ LoginLoA_{UserIdP}]$$

The relying party should be sent the registration LoA and login LoA as part of the authentication assertion, so that it can ensure that neither exceed the trusted maximum values, and it can then compute the session LoA itself.

The third proposal is to allow users to login to different IdPs within a single session, to prove that they are the owner of a set of IdP accounts. This account linking procedure must be managed by a trusted service provider (TSP) which remembers the set of accounts that the user has linked together, and the session LoAs associated with each one. From these session LoAs it dynamically computes a registration LoA for the user with itself ($RegLoA_{user}$) based on the highest of the individual session LoAs, viz:

$$RegLoA_{user} = Max\ [SessionLoA_{UerIdP1}\ ..SessionLoA_{UerIdPn}\ ]$$

The reason that the session LoA is used to compute the registration LoA (and not the individual $RegLoA_{UserIdPi}$) is that a strong registration procedure is weakened by a weak login process so that a remote user can only be assured to the combined value of the two. The TSP stores the user's registration LoA with itself. Note that the user is free to break any account linkage at any time by asking the TSP to delete a particular account from his linked set, in which case the user's registration LoA will be recomputed from the remaining accounts. A user is also free to re-authenticate with the same or a different IdP using an alternative authentication method, in order to try to increase his session LoA with that IdP, and thereby increase his registration LoA with the TSP.

The fourth proposal is that when a user logs in to an SP, the SP routes this request via the TSP, which acts as a proxy IdP for the SP. The LoA for the user's session with the SP is dynamically computed by the TSP from the stored registration LoA of the user and the login LoA of the user with his chosen IdP, viz:

$$LoA_{session} = Min\ [RegLoA_{user}|LoginLoA_{UserIdP}]$$

In this way we can leverage the strongest linked registration procedure undertaken by the user, with the current SSO login process to potentially compute a higher session LoA thereby giving the user increased privileges with the SP.

3.2 Trust Model

The TSP has to be trusted by the SPs and users that use it. For this reason the TSP should be part of the SPs' domain and managed by it. Users who contact this organisational domain for various services will be given the option of linking their various external IdP accounts together via the organisation's TSP. Each time a user tries to access a service within the organisation, she will be routed via the TSP when the SP asks her to authenticate via her preferred IdP. The various IdPs are required to trust the TSP as a service of the organisation, and do not need to know that the TSP is a proxy for other services of the organisation.

3.3 Attribute Model

A user may have different attributes asserted by different IdPs. A user may also have attributes stored in the organisation's corporate server (we currently assume this is an LDAP server but it need not be). The SPs will always trust the attributes provided by their own corporate server to be asserted at any LoA. However, the SPs will only trust the attributes provided by a remote IdP if they are asserted with a session LoA greater or equal to a requested threshold. When a user has linked his corporate account to one or more remote IdP accounts, and the user logs in via one of these remote accounts, then the TSP will always add the user's local LDAP attributes to its response to the SP (to match the SP's requirements). However the TSP will only keep any remotely asserted attributes if the IdP's session LoA is equal to the user's computed session LoA, i.e.

$$SessionLoA_{UserIdP} = LoA_{session}$$

otherwise they will be discarded from the TSP's response to the SP. This is because the user's session LoA may be increased by the TSP, and we do not wish the user to be able to assert remotely assigned attributes at a higher LoA than they deserve (even if the assurance of the user's authentication is increased).

4. ARCHITECTURAL DESIGN

The TSP comprises a user account database (the Account DB), an Account Management SP and a proxyIdP. The user account linkages and computed registration LoAs are stored in the Account DB which is managed by the Account Management SP. The LoAs (registration and login) of the known and contactable IdPs, are read in at start up from either the Federation Metadata or a configuration file.

The Account Management SP is responsible for calculating the user registration LoA ($RegLoA_{user}$) dynamically as the user links (and unlinks) his various accounts together. When a user wishes to add an IdP account to his linked set, via the Account Management SP, he is first redirected to the proxyIdP, which allows him to choose between the contactable IdPs. Once the user has been authenticated by an IdP, the returned details of the IdP are stored in the Account DB by the proxyIdP and the user's registration LoA is then computed from the current set of linked accounts by the Account Management SP.

The primary tasks of the proxyIdP are to:

- act as an IdP discovery and filtering service by prompting the user to select his preferred IdP from the set that matches the SP's requirements, in terms of both a required LoA and set of attributes,
- compute the session LoA ($LoA_{session}$) in its response to the SP (excluding the account management SP) based on the stored registration LoA and the current login LoA,

- be a protocol gateway to communicate with the various IdPs, and
- determine which user attributes are placed in the response to the SP.

The SPs communicate with the proxyIdP using a standard protocol (we use SAMLv2), and the proxyIdP communicates with the various IdPs using their supported protocols e.g. SAMLv2, LDAP, OpenID, Oauth, Facebook etc. New protocols can be added to the proxyIdP as they become available. LDAP is used to communicate with the local organisation's LDAP service for users who have local accounts with the organization. The LDAP service is assumed to hold the users' login credentials and local attributes. LDAP can therefore act as the local IdP.

The user chooses which IdP he wants to use for login and the proxyIDP then communicates with it using the appropriate protocol. It maps the response into a SAMLv2 assertion which it sends to the requesting SP.

The overall architecture is shown in Figure 1.

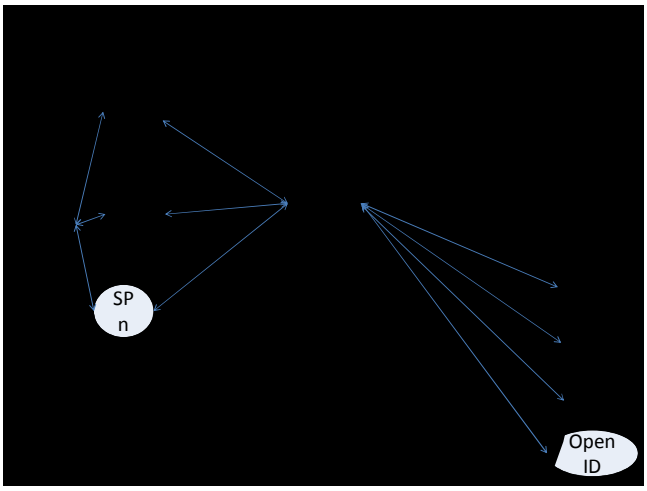

Figure 1. Overall Architecture

A user may access the organisation's public services (represented by Public SP in Figure 1) without any restrictions. However, whenever the user attempts to access any protected service in the organisation (SP1….SPn in Fig. 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 as well as its required attributes. This causes the proxyIdP to tailor the login screen to only include those IdPs which are able to equal or surpass the requested LoA (based in their configured trusted LoA_{IdP} values) and published attributes (after discounting any attributes that can be obtained from the local LDAP) i.e. the proxyIdP knows the maximal values that these IdPs can issue, but does not know whether they will issue them for this particular user. This filtering is necessary but not sufficient i.e. there is no point in including IdPs in the list which the proxyIdP knows cannot fulfill the SP's requirements, but until the user has been authenticated the proxyIdP cannot know for sure which IdPs are capable of fulfilling them.

The account management SP is made available at the organisation's web site in the same way as any other SP, so the user does not need to do anything special to invoke it in order to link together his IdP accounts.

4.1 Account Linking

Before a user is known to the TSP, she will not have any records in the Account DB, regardless of whether she exists in the organisation's LDAP service or not. The first time a (unknown) user interacts with the TSP and chooses an IdP to authenticate to, the proxyIdP will receive a persistent identifier (PId) from the IdP. The PId might be a username (e.g. from Twitter), or a uni-directional identifier (e.g. from a SAML IdP). The proxyIdP searches for the IdP/PId combination in the Account DB, 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 organisation's SPs, in which case this is all that will ever be stored in the Account DB.

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

Whenever a user chooses to authenticate via the organisation's LDAP, this causes the proxyIdP to store, in addition to the IdP/PId, the user's LDAP DN in the Account DB. Thereafter the proxyIdP is able to retrieve the user's local attributes.

The user can choose to link together her existing IdP accounts whenever she chooses via the Account Management SP. Access to the Account Management SP is in the same way as to all the other SPs and requires the user to login via the proxyIdP. Once authenticated, the user is shown her currently linked account details, which first time around will only be the current IdP account. The user may choose to link any of her other IdP accounts to this one by selecting the Add New Account option, in which case she is logged out of the current IdP by the Account Management SP and redirected to the proxyIdP which displays the IdP selection screen again. The user chooses a new IdP, is redirected to it, authenticates and is then redirected back to the proxyIdP. If the returned PId-IdP tuple is unknown to the proxyIdP, it inserts a new account entry in the Account DB. The user is then redirected back to the Account Management SP for account linking to take place and the Registration LoA to be recomputed. If the newly logged in account is not already linked with any other accounts then it is simply added to the current set of the user's linked accounts, but if the account is already linked with other accounts the user is given a Warning message and asked if she wants to merge this account (and all of its existing linked accounts) with the current set of account(s), or move the account on its own to the current set, or cancel the operation without linking it to the current set. In this way the user is left completely in charge of managing her identity, and can have as many sets of linked and unlinked accounts as she wishes in the TSP.

The Account Management SP allows the user to link/unlink different accounts with each other and thereby increase/decrease the registration LoA value associated with each set of linked accounts.

4.2 Authorisation

Authorisation comprises two parts: firstly the proxyIdP retrieves the attributes and LoA of the user, and secondly the SP makes an access decision based on these, typically by calling a backend authorisation server to obtain an access decision. The proxy IdP gets the user's attributes from up to two sources: i) if the user has a linked local account, the LDAP server, and ii) from the authenticating IdP if its session login LoA is equal to the

computed session LoA i.e. $SessionLoA_{UserIdP} = LoA_{session}$. The latter stops an unsafe escalation of assurance of externally asserted attributes. For example, users who might have linked two external accounts together, one with a high LoA that asserts no attributes, and one with a low LoA that asserts many attributes, are not able to increase the LoA of the asserted attributes. They can either choose to authenticate with an unlinked IdP with a low LoA which asserts the attributes, or with a linked IdP to obtain a higher LoA but without the attributes. On the other hand, users who are registered with the organization and have attributes in the corporate LDAP (which are always trusted), 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 above. In this way the proxyIdP can still retrieve their attributes from the organisation's LDAP server even when an external IdP was used.

Once the proxyIdP has obtained the user's initial set of credentials it can determine if they are sufficient to fulfill the SP's requirements. If they are not, it returns the user to the IdP discovery screen with an improved filtered list of IdPs which has now been tailored to the specific user, and the user will have to authenticate again to a different IdP in order to fulfill the SP's requirements.

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. 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's session. 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 organisation's LDAP server or an external IdP.

5. IMPLEMENTATION

5.1 Determining the LoAs

In order to implement the design, we have to determine

a) the LoA of each IdP (i.e. the components of LoA_{IdP}) which represents the maximum trust we can have in an IdP

b) the LoA of each user session (i.e. the components of $SessionLoA_{UserIdP}$) which should not exceed the maximum above

For SAML based IdPs, the SAML community has recently published a committee draft [11] that defines an attribute (urn:oasis:names:tc:SAML:attribute:assurance-certification) suitable for inclusion in SAML Metadata which provides the IdP's assurance certification i.e. LoA_{IdP}. The Shibboleth implementation of SAMLv2 has recently introduced support for this. Whilst this is only intended to publish a single LoA value, the schema allows it to contain multiple values, so an IdP can use it to publish both its registration and login LoA values.

SAML IdPs that do not yet support this standardised feature have often had other non-standardized mechanisms for indicating an IdP's LoA. For example, the UK Access Management Federation (AMF) agreement [13] contains clause 6 which states that IdPs which can "match the use of services provided by Service Providers to individual End Users" can publish this declaration in their Metadata. This equates to them saying that they conform to a registration LoA of at least 2. So it is possible to determine whether a UK AMF IdP's registration procedures are to LoA 2 or not by looking at the appropriate field in the federation metadata. However there is currently no way of knowing what their login LoA is.

The fallback position is that the SP can ask each IdP that it trusts for its registration and login LoA values when setting up its contractual agreements with them.

Dynamically determining the session LoA is more problematic. The OASIS draft [11] provides guidelines for the use of SAML's Authentication Context mechanism (which uses the <AuthnContextClassRef> element) to request and express an LoA in SAML assertions (i.e. $SessionLoA_{UserIdP}$). Extending this to return both the registration and login LoAs is inconvenient as it means combining two values into one, but this is not impossible. It can be done by defining m*m LoA URIs rather than 2*m URIs, where m is the number of LoA values (4 in the NIST case). However, no existing UK AMF SAML IdPs currently return any LoA information dynamically, so in the current implementation we have had to assume that the dynamic values are the same as the statically configured maximum values.

LoA information is neither published nor dynamically transferred for SN and other non-SAML IdPs, so we have had to determine these by experiment. In order to do this, we created pseudonymous accounts on Google, Facebook, Twitter and an OpenID provider in order to see what registration information was asked for, how it was validated, plus what the minimum requirements for password strength were and how easy it was to brute force crack them. We then computed the IdP's LoAs from the experimentally determined registration and login LoAs. We further assumed that the dynamic session LoAs will always be the same as these experimentally determined and statically configured values.

The experimental results are as follows. Google validates the email address of the user, so we had to register the email address csidiot@yahoo.com in order to open a Google account. No other details were validated. So this would only gain the lowest registration LoA of 1. The password field on the registration form specifies that the password must be eight or more characters in length. There are no other stated requirements although we were not allowed to use "password" or any dictionary word that we tried. A password strength meter records the strength and won't allow weak passwords to be used. Using Table A.1 of [8] we compute this as 30 bits of entropy. Concerning wrong password attempts, after 6 wrong passwords, you are required to complete a Captcha puzzle on every subsequent attempt. The best we could attempt was 25 wrong passwords in 5 minutes by clearing cookies after each 6 wrong passwords. This equates to 2.7M attempts per year (approx. 21 bits of entropy). So Google just about qualifies for Login LoA 1, which requires 10 bits of entropy for the life of the password.

Twitter allows the user to register any name (we used idiot) and does not check any of the details including the email address – we used idiot@spam.la and it still created the account. This only qualifies for the lowest registration LoA of 1. The only restriction imposed on the password was that it must be six or more characters in length and "passwd" was allowed. It appears that any dictionary word is allowed so this only scores 14 bits of entropy from NIST's Table A.1. After 3 false password attempts you are required to complete a Captcha with every password attempt. However if you simply replace the URL https://www.twitter.com/#!/login/captcha/ with https://www.twitter.com/#!/login/ then no Captcha is asked for.

Consequently we were able to perform 100 login attempts in 29.9 seconds with JMeter. At this rate 105M passwords could be tried in a year (approx. 30 bits of entropy) and it would take just over a day to crack a weak password. This does not remotely qualify for NIST login LoA 1 and must be given 0.

Facebook has some restrictions on first and last names, since idiot was not allowed for either. However no registration details are validated as we used an email address of idiot@<a valid domain> and this was allowed even though it does not exist. Again this qualifies for the lowest registration LoA of 1. The password must be at least six characters in length, and like Google it does not specify any other requirements but it will not allow dictionary words. It did however allow "facepass". We gave this 23 bits of entropy from Table A.1. After 3 wrong password attempts the user is given a Captcha. The user can either clear out cookies and try another 3 times, or complete the Captcha and then continue. After each 3 wrong attempts the user is sent via several redirects to "help" pages, which in fact slow down an attacker. After 33 false passwords we were given an error message about too many failed attempts and then had to complete some more pages which takes another minute or so. The best we could achieve was 33 wrong passwords per 6 minutes, which is 2.9M per year (approx. 21.5 bits of entropy). So whilst it would take over a year to crack a weak password with Facebook, it does not quite qualify for a login LoA of 1 and unfortunately we have to give it a 0.

Since there are many OpenID providers it is impossible to determine a generic registration process as it varies with the provider. We registered on myid.net with idiot.myid.net and the email address of idiot@spam.la. No validation was performed. This still qualifies for the lowest registration LoA of 1. The password must be six characters or more in length, but no other restrictions apply and we registered with "password". We gave this an entropy of 14 bits. Multiple false passwords can be tried repeatedly with no restrictions. Using JMeter we tried 1500 passwords in 2mins 30secs, which equates to 315M in a year (approx. 31.5 bits of entropy). This is clearly a login LoA of 0. In order to qualify for login LoA 1, this OpenID provider would need to request 25 character long passwords from its users.

By way of comparison, the University of Kent performs face to face registration of both staff and students that requires the person to present their passport. In addition students have to provide their degree certificates, and staff their national insurance details. This qualifies for a registration LoA 4. It's password policy states that passwords must be between 9 and 16 characters chosen from at least three of: upper case, lower case, numerals and special characters. This automatically rules out dictionary words and gives an entropy of 31 bits. Passwords have to be changed at least every 9 months, and recently used passwords cannot be used again. After 3 wrong password attempts the user is redirected to another page before they can try again. We managed to enter 9 wrong passwords in one minute. After 10 wrong passwords the user is locked out for one hour, so the most password attempts one can have in its lifetime is 65K, or 16 bits of entropy. This leaves 15 bits of entropy which exceeds NIST's requirements for login LoA 2.

The experimentally determined values for the registration and login LoAs for the above IdPs is stored in a configuration file and read into the TSP at initialization time. We might add that the login LoAs we have assigned to the above SN IdPs are for the worst case scenarios and apply to users choosing the weakest passwords possible. If the IdPs change their login processes these values will need to be re-assessed. But in order to protect the organizational resources we have to take the most cautious approach.

5.2 Proxy IdP

The proxy IdP has been implemented using the open source SimpleSAMLphp [14]. 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, LDAPAuth 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 proxyIdP. The first, loaauth, allows the set of IdPs to be filtered according to the LoA requested by the organisation's SP in the <authnContextClassRef> inside the SAML authentication request. This ensures that the 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. We are currently in the process of adding attribute filtering to this as well, so that IdPs that cannot provide the required attributes (over and above the ones provided by the organisation's LDAP) are also filtered out.

The second, loaldapcheck, computes the correct LoA and set of attributes to be inserted into the response to the SP. First it checks if the authenticated user has an entry in the Account Database, and if not creates one. Next it checks if the user has a linked account in the organisation's LDAP. If so, it retrieves the user's attributes according to its preconfigured Attribute Release Policy (ARP) and the SP's request, and inserts these into the response to the SP. It then computes the user's session LoA from the stored registration LoA and the current login LoA and adds this to the user's set of attributes (the rationale for this is explained in the next section). Finally it removes any IdP asserted attributes if the computed session LoA is greater than the configured LoA of the IdP.

The simpleSAMLphp code then creates a signed and encrypted SAMLv 2.0 <samlp:Response> message for the requesting SP, containing the computed LoA and retrieved attributes.

5.3 Protocols

We use the SAMLv2 protocol between the organisation's SPs and the proxyIdP and between the proxyIdP and the SAML IdPs. The SP use the <AuthnContextClassRef> to request the LoA from the proxyIdP. The SP uses the open source simpleSAMLphp code to process the SAML requests/responses with the proxyIdP. Unfortunately this code does not provide a mechanism to extract the <AuthnContextClassRef> element from the response message, in order to retrieve the user's session LoA. 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. Consequently this way was deemed to be unsuitable;

ii) we could modify the proxy IdP to add the session LoA value to the set of subject attributes in the attribute assertion of the

SAMLv2.0 response it creates. The SP will then transparently handle the LoA attribute along with all the other user attributes. This is the approach we adopted.

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 published in the SP's metadata. Consequently, we recommend that the SAMLv2.0 specification be enhanced 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. In the current implementation the proxyIdP (acting as the organisation's SP) sets the <AttributeConsumingServiceIndex> attribute to the value 0, which means return ALL attributes, then the proxyIdP does the attribute filtering.

For non-SAML IdPs, the proxyIdP utilises the appropriate protocol module to access the user's chosen IdP. The latter performs user authentication and returns an authentication assertion to the module along with the user's attributes (if any). The protocol module validates the received assertion, extracts the PId/IdP tuple and any attributes, appends the configured session LoA for this IdP as experimentally determined above, then returns these to the proxy IdP. The proxy IdP then creates the <samlp:Response> message for the SP.

When the SP receives the <samlp:Response> message from the proxyIdP 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. If the response is valid, each user attribute contained in the assertion (including the session LoA) is parsed and mapped into its equivalent XACML counterpart and added to an XACML request context. The PDP policy can then treat the LoA attribute as just another user attribute when making an authorisation decision.

The SP now talks to the backend authorisation server using the SAML-XACML protocol [12]. This request contains a standard XACML request-response context [10] wrapped as a new type of SAML assertion – the XACMLAuthzDecisionQuery assertion. The SAML-XACML protocol allows additional levels of security to be added to the request, such as authentication and encryption. In our implementation we use TLS. It also allows the SP to specify the authorisation policy to be used by the authorisation server to process the SP's request. This allows us to support multiple PDPs and multiple policies in the authorisation server, and to 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, such policy to be used to authorise the user's request.

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

6. 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 three tasks, and to "think aloud" whilst doing so. This allowed the observer to tape record the user's thought processes for later analysis so as to discover where any usability problems might lie.

These first set of tests were performed with our initial prototype system that allowed the proxyIdP to filter suitable IdPs based on their LoAs, but not on their attributes. This feature is currently being implemented.

6.1 User Tasks

The users were asked to perform 3 different tasks.

1. Download the University of Kent postgraduate application form.

This task requires an LoA of 0 which can 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. The user can subsequently track their application through the system by logging in using the same (or another linked) IdP (not part of this task).

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

This task requires the user to have an LoA of at least 1 and an eduPersonAffiliation of "University of Kent". The LoA is achievable by either logging in via the university LDAP, or via most UK-AMF IdPs or Google or Facebook (which we configured to have a login LoA of 1 for these tests). The eduPersonAffiliation can only be obtained by either logging in via the university LDAP, or one of the user's other LoA 1 accounts which has previously been linked to the user's LDAP account.

3. 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 LDAP or the University of Kent IdP in the UK-AMF.

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 given the un/pw for a UK-AMF account with an LoA of 2, and asked to use this account first and then any other account of their choosing.
5. User 5 was the same as User 4, but after performing the 3 tasks was specifically asked to link his accounts together and then repeat the 3 tasks again.

6.2 User Trials Results

The tests were designed so that all users, except User 2, would be able to complete all three tasks. User 2 would only be able to

complete Task 1. These results were indeed obtained, as expected. However, these 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 most cases the users proceeded without difficulty and were able to complete the tasks in a very straightforward manner meaning that few comments were recorded. A breakdown of each user experience is given below.

User 1 - used his[2] 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 he 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. He noted that the tasks were easy to complete, and that the system was simple to understand but it would have been helpful to know which account he was currently logged into, and that perhaps the pages should display a message such as "Logged in via Facebook".

User 2 - was asked to login using his Facebook account and was therefore unable to complete any tasks other than task 1. Despite this the user noted that he understood why access was being denied as he had only used Facebook to log in.

User 3 - was asked to use his 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.

User 4 - was given account details for a UK-AMF LoA 2 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 of Kent 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.

User 5 - was given the same instructions as User 4 but after he had completed the 3 tasks he was asked to access the Account Management service and link some of his accounts together, then log out and repeat the tasks again. The results were interesting. Task 1 was completed as per User 4, but when User 5 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 our original "attribute based IdP filtering" design issue which is now being implemented. User 5 then chose his university account and successfully accessed Moodle and completed the remaining task. 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). 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 another symptom of the attribute based IdP filtering problem discussed above. He said aloud that this was because the SDS was only accessible to University of Kent 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 5 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 to his other ones..

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 5) 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 to the system. 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.

7. Discussion

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 Google (or Facebook for this test), 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 and) Google. 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.

Clearly one major issue is that Facebook's login process is still not robust enough to qualify for LoA 1. (We set it to 1 for these tests to see how users would adapt to using it to access university resources.) Whilst most Facebook users may well choose medium or strong passwords which will pass the LoA 1 threshold, some will not. So for this reason Facebook accounts won't be able to be used to access privileged university resources until their login process is strengthened.

[2] The gender in the text does not necessarily reflect the gender of each participant

Another major issue is that most IdPs do not currently reveal their overall LoA let alone the two major components that comprise it. Until IdPs are willing to be publish their LoAs and transfer them in protocol, the scheme proposed here will need significant effort by SPs to determine the LoAs of their "trusted" IdPs.

One outstanding issue at the time the trials were run was that of IdP filtering based on the attributes they release (which User 5 encountered). Since our original prototype we have improved the design (as described in this paper) so that the proxyIdP will request and store a list of attribute types that each IdP is prepared to release for each user. The will allow the proxyIDP to filter out inappropriate IdPs and allow the user to select between the remaining ones based on the attributes and LoAs the IdPs can provide. This is currently being implemented.

One major issue that currently has not been addressed is how to retire dead accounts. We expect that the Account DB will grow to many hundreds of thousands of entries over time, and that some users will lose interest, 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 [15] is starting to attract significant attention in the identity management world.

There are many as yet unexplored opportunities for leveraging this system. Password management is one of them. This is currently 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 with as strong as or stronger login LoA, it should be possible, after logging in via the external account, to request that a new university password be emailed to them. Providing the Login LoAs of the external account and the email system are as strong as the university's login LoA, then the same level of assurance will have been obtained in order to allow a new password to be issued, as a change password request today.

The developed software is being released as open source code through the university's web site and Feide, the authors of simpleSAMLphp. A public demonstration (of the system used in the user trials) is also available [16].

8. ACKNOWLEDGMENTS

The authors would like to thank the UK JISC for funding this research under their Access and Identity Management Program and the EC for funding the current enhancements under the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 216287 (TAS³ - Trusted Architecture for Securely Shared Services).

9. REFERENCES

[1] 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.

[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] 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] E. Hammer-Lahav, D. Recordon, D. Hardt "The OAuth 2.0 Authorization Protocol". draft-ietf-oauth-v2-18. 8 July 2011

[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 2008Bowman, M., Debray, S. K., and Peterson, L. L. 1993. Reasoning about naming systems. ACM Trans. Program. Lang. Syst. 15, 5 (Nov. 1993), 795-825. DOI= http://doi.acm.org/10.1145/161468.161471.

[9] David W Chadwick, George Inman. "Attribute Aggregation in Federated Identity Management". IEEE Computer, May 2009, pp 46-53

[10] OASIS "eXtensible Access Control Markup Language (XACML) Version 2.0" OASIS Standard, 1 Feb 2005

[11] OASIS "SAML V2.0 Identity Assurance Profiles Version 1.0" Committee Specification 01, Nov 2010. Available from http://wiki.oasis-open.org/security/SAML2IDAssuranceProfile

[12] OASIS "SAML 2.0 profile of XACML, Version 2.0". OASIS committee specification 01, 10 August 2010

[13] UK Access Management Federation for Education and Research. "Rules of Membership" 1st November 2007. Available from http://www.ukfederation.org.uk/content/Documents/FedDocs

[14] SimpleSAMLphp is available from http://simplesamlphp.org/

[15] See http://digitaldeathday.com/ (last accessed 9 March 2011)

[16] Demonstration available at https://persistence.kent.ac.uk/logins4life/