Attribute Aggregation in Federated Identity Management

David W Chadwick and George Inman, University of Kent

Abstract

We describe how in today's federated identity management (FIM) systems, such as CardSpace and Shibboleth, service providers (SPs) rely on identity providers (IdPs) to authenticate the users and provide their identity attributes. The SPs then use these attributes for granting or denying users access to their resources. Unfortunately most FIM systems have one significant limitation, which is that the user can only use one IdP within a single SP session, when in many scenarios the user needs to provide attributes from multiple IdPs. We describe how this can be achieved through the introduction of a new service called a *linking service*. The conceptual model of the linking service is described as well as the mapping of its messages onto today's standard protocols (SAML, Liberty Alliance and WS-*).

Comment [DWC1]: For a side bar

Definitions

Attribute – a distinct characteristic of a person.

Attribute Authority (AA) – an entity that is trusted to assert that a particular person has a particular attribute

Authoritative Source – an AA that can definitively answer queries concerning a specific identity attribute for which it is responsible.

Identity – a set of attributes that characterise a person.

Identifier – A string which is used to uniquely identify an entity within one domain or system.

Identity Provider (IdP) - a type of SP that creates, maintains, and manages identity attributes for people and provides user authentication. An IdP is an AA combined with an authentication service.

Service Provider (SP) - an organisation that offers web based services.

Introduction to Federated Identity Management

No single person or system knows anyone's complete set of identity attributes. Individuals are most likely to know the majority of the attributes that serve to identify them. But even then there are limitations, for example, individuals might not know how much others trust them. Invariably then, computer systems typically only hold the partial identities of people i.e. a subset of their digital identity attributes. These computer systems are known as identity providers (IdPs). Usually identity attributes have to be conferred on individuals by their authoritative sources. Whilst people may be trusted to

assert some of their identity attributes, for example, their favourite drink, they certainly wont be trusted to assert all of them, for example, their qualifications or criminal records. Thus different authoritative sources are responsible for assigning different attributes to individuals.

Access Control in FIM

In the role based access control (RBAC) and attribute based access control (ABAC) models, authorisation is based upon the user's roles or attributes respectively. Federated identity management systems typically adopt the ABAC model. The systems that assign attributes to users, i.e. the IdPs, are different from and remote to the systems that consume them and grant access to the users i.e. the service providers (SPs). Thus trust needs to be established between the IdPs and the SPs. Federations are built upon this trust. Authorisation to use a federated service is granted based on a user's identity attributes. If the latter are provided by trusted authoritative sources, then the SP can be assured of the identity of the user, even if the user's identifiers at the various IdPs are unknown to the SP. Hence systems such as Shibboleth [1] and CardSpace were born. These allow an arbitrary number of IdPs and SPs to form trust relationships between themselves in a federation.

Unfortunately, one of the current limitations of Shibboleth, CardSpace and similar systems is that the user can only select one of his IdPs in any given session with a SP. The user selects one of his IdPs, is authenticated to it, and the IdP then sends an authentication and attribute assertion to the SP. Consequently authorisation is limited to a subset of the user's identity attributes. For many web based services this is not enough. We need a mechanism to allow a user to select (or aggregate) attributes from multiple IdPs in a single service session. In short, attribute aggregation is required.

Previous Research in Attribute Aggregation

Early work on merging or aggregating attributes from multiple attribute authorities assumed that the user had a globally unique identifier which was common across all the attribute authorities [2]. This name identifier was usually an X.500 distinguished name held in an X.509 public key certificate assigned by a Certification Authority. This was subsequently standardised in 2001 in X.509 attribute certificates. The user only needed to authenticate once, with his public key certificate, and then merging together the different attributes from the different attribute certificates was easy since they all contained the same user distinguished name.

In reality, few users have X.509 public key or attribute certificates, and instead have different user names and attributes assigned to them by their various IdPs. The Liberty Alliance was the first group to address the attribute aggregation problem in this scenario, through their concept of identity federation [3]. In this model, as the user is moving between services in a federation, the first IdP to authenticate the user asks him if he would like to be introduced to other IdPs in the federation. When the user subsequently authenticates to a second IdP, it invites him to federate his second identity with that from the first IdP. If the user agrees, this causes the two IdPs to each create a random alias for the user and to exchange these behind the scenes. In this way, neither IdP knows the true

login identifier of the user at the other IdP, but each may refer to the same user via the random aliases, and thereby aggregate the attributes. Whilst this model is good at protecting the privacy of the user's identifiers, and stops IdPs from exchanging data about users without the users' consent, each IdP still knows that a "federated" user has some attributes at the other IdP. This is not mirrored in real life. There is no reason for my credit card company to know that I am a member of IEEE, or vice versa, yet I might still want to use both these attributes in a single transaction, for example, to buy a book from an online store and gain a discount due to my IEEE membership. Thus a SP has to be given the aggregated set of attributes from multiple IdPs, but without the IdPs knowing about each others involvement.

Chadwick built on the Liberty Alliance work in [4]. This model envisaged IdPs forming pairwise relationships, called partnerships, and sharing secret keys between themselves to cement their relationship. At any time a user could link together his two accounts at a partnership by authenticating to each IdP in two separate web browser sessions and providing each IdP with the same random string. They could then transfer this secret to each other, thereby providing proof it was the same user simultaneously accessing them both. This enabled them to link the user's two accounts together, with each including its own random alias in the message exchange. When a user subsequently contacted a SP and the SP requested the user's attributes from one IdP, the IdP would return the random alias (suitably encrypted) and details of the partner IdP to the SP, allowing the SP to fetch the linked attributes. Whilst this scheme has the same privacy properties as the Liberty Alliance model, it also suffers from the same privacy deficiency.

MyVocs [5] developed an alternative mechanism based on a model that Klingenstein calls Identity Proxying [6]. In this model the SP has one IdP that it trusts absolutely. Other IdPs are unknown to the SP and they only have trust relationships with the primary IdP and not with the SP. All user access requests are channelled through this trusted IdP, which then relays the user to his chosen IdP. The chosen IdP authenticates the user and returns his attributes to the trusted IdP, which strips off the assertion wrapper, supplements the attributes with its own user attributes and returns the aggregated set to the SP as its own attribute assertion. The danger with this model is that the trusted IdP can assert any attributes about any user to the SP, since it is trusted absolutely to assert everything. Knowledge about which IdPs originally asserted which user attributes is lost to the SP. This trust model will not work in many real life scenarios.

In his review of the various attribute aggregation models, Klingenstein [6] also describes:
- the applications database model, in which an SP stores a subset of user attributes locally and merges these with ones provided by a federated IdP;
- identity relay, an advanced form of identity proxying which reduces the amount of trust that is needed in the relaying IdP. In this model the SP receives attribute assertions from both IdPs rather than from just the relaying IdP;
- the client mediated assertion collection model, in which an intelligent client independently guides the user to authenticate to multiple IdPs, pulling attribute assertions from each one, and then presenting the combined set to the SP,
- IdP mediated attribute aggregation, which is the model of Chadwick described above,

- SP mediated attribute aggregation in which the SP sequentially redirects the client to different IdPs. This requires the user to authenticate to each IdP in turn and retrieve an attribute assertion from it which is then returned to the SP. The SP continues to collect the various attribute assertions until it has enough to authorise the user.

Whilst SP and client mediated collections are secure and fully protect the privacy of the user as there are no links between the IdPs, their downside is that the user has to authenticate to each IdP in order to collect the various attribute assertions. Users may find this overhead too tedious. Identity relay is secure but compromises the user's privacy somewhat in that the relaying IdP is aware of the user's links with the other IdPs. This violates Kim Cameron's 3rd law of justifiable parties (see http://www.identityblog.com/?p=352/#lawsofiden_topic). The model we propose is a variant on both the IdP mediated attribute aggregation and identity relay models and introduces a *linking service* to both hold the links between user identities and relay attribute requests between SPs and IdPs.

Attribute Aggregation Conceptual Model

Our conceptual model for attribute aggregation assumes that the user is the best (and probably only) person to know the authoritative sources for his identity attributes. This is a reasonable assumption to make, since most people know who issue their credit cards, passports, health cards, driving licenses, group memberships etc. We also know that privacy protection is important from a requirements survey that we carried out prior to the design of our system [7]. We have therefore devised a new web service, called a *linking service*, whose purpose is to hold links between the user's various IdPs, as directed by the user, whilst fully preserving the user's privacy. Privacy preservation is achieved in the following way. Whilst each IdP knows one partial identity of the user, no IdP is aware of any of the user's other partial identities. Whilst the linking service knows that a user has several linked partial identities, it does not know any of them or who the user is, since it delegates user authentication to the linked IdPs. Consequently the linking service only knows that some user is known to several IdPs, and it holds the links to these on behalf of the user, without knowing who the user is.

When the user contacts a SP for service provision and is redirected to his IdP for authentication, the IdP returns a pointer to the linking service in its response. This allows the SP to contact the linking service in order to achieve attribute aggregation. The linking service may either relay the SP's request to each linked IdP, and relay the encrypted responses back to the SP, or it may return the set of linked IdPs to the SP allowing it to aggregate the attributes. The linking service is under the total control of the user, who creates and deletes the links, and says which linked IdPs can be released to which SPs, through an IdP link release policy (see sidebar).

Privacy preservation is ensured through a minimal of trust. The user, IdPs and SPs trust the linking service to hold the IdP links securely, and to only divulge them to SPs under the instructions of the user. The linking service is simply trusted as an honest broker to

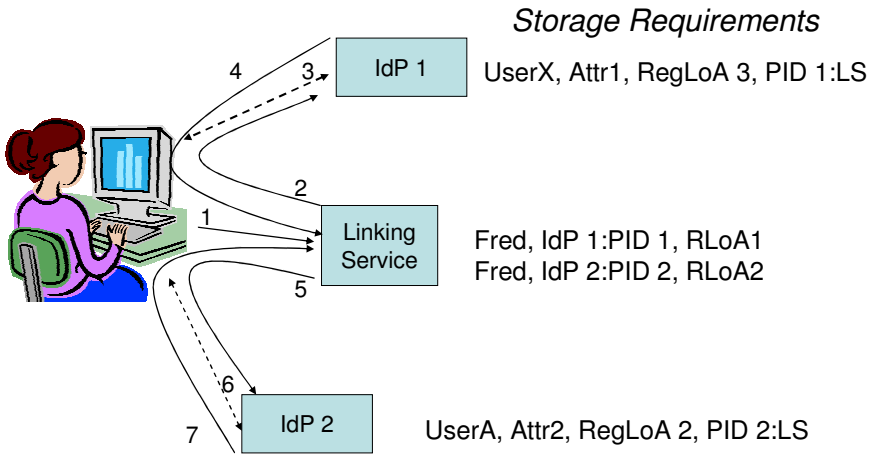
keep the links secure without knowing the identities of any of the users. If an IdP or SP does not trust the linking service it will simply not interact with it.

Our model removes the burden from the user of having to authenticate to each IdP separately during service provision. Only one authentication exchange is required, and this can be to any one of the IdPs linked together in the linking service. The following sections describe in more detail how the linking service works.

Link Registration Phase

The user goes to the web page of his preferred linking service (there can be any number of these on the web, or it could run on the user's own PC). The linking service displays a list of all the IdPs with which it has trust relationships. The user selects one that he wants to create links to. The linking service redirects the user to the chosen IdP, whereupon the user is asked to login and authenticate. The user authenticates using the IdP's chosen method. Upon successful authentication, the IdP creates a random (but permanent) identifier for the user which is to be used solely with the linking service. The IdP returns an authentication assertion containing this permanent ID. This assertion effectively says "I have authenticated this user, and he is to be known by you as PIDx." When the linking service receives this message it creates a new link entry for the user in its linking table, assigning the user its own local identifier, say Fred, then displays the list of IdPs again. The user selects another IdP, is redirected there, authenticates, and the second IdP returns a different permanent identifier, say PIDy, to the linking service. The linking service adds this link entry to its linking table. The user can perform this IdP linking as many times as he wishes, and the linking service will create a new link table entry for this user each time, as in Table 1. The linking process is shown pictorially in Figure 1.

Each PID is regarded as a secret between the linking service and the issuing IdP and therefore must be encrypted with the public key of the recipient when being transferred between the two.



1. User contacts her preferred linking service, and chooses one of her IDPs (IDP 1)
2. User is redirected to her chosen IDP
3. User authenticates to IDP, and IDP generates a PID for this user with this linking service (PID 1:LS)
4. User is redirected back to linking service with an authentication assertion that carries the Session LOA and PID. Linking service stores this in the user's linking table entry.
5. User chooses another of her IDPs and the whole process is repeated again

Figure 1. Establishing links between IdPs

Level of Assurance

Different IdPs will authenticate users in different ways, and to different strengths e.g. usernames and passwords are weaker than public key certificates and private keys. This is termed the Level of Authentication, or Level of Assurance (LOA). It is the assurance that a relying party can have that the user is really who it thinks he or she is. The assurance a relying party has, depends not only upon the electronic authentication method that was used, but also on the initial registration process that preceded this, for example, registering electronically over the web is much weaker than turning up in person with your passport. NIST has a recommendation which classifies the LOA at four levels, with level 4 being the strongest and level 1 being the weakest [8]. Some SPs may wish to grant a user different access permissions based on the LOA of their current session, e.g. if the user authenticates with an LOA of 1 they can read the resource, but with an LOA of 3 they can modify its contents. One of the limitations of the NIST recommendation is that the LOA is a compound metric dependent upon both the strength of the registration process and the strength of the electronic authentication method. We believe it is more useful if they are separate metrics, as described below.

Prior to any computer based authentication taking place, a user needs to register with a service, and provide various credentials to prove their identity. For example, before a new student is registered to use the University of Kent's computing services, they must first present their passport and existing qualifications, to prove they are entitled to register as a student. We call this the *Registration LOA*. Different systems will require different registration documents and have different registration procedures, and will therefore have

different Registration LOAs. After successful registration, the university allocates the student a login ID (their identifier) and associates various attributes with this in its database, e.g. degree course, student's name, date of birth, email address, department, tutor and so on. The university may offer different authentication mechanisms for student login, such as un/pw with Kerberos, un/pw with SSL, one time passwords via a mobile phone etc. Each of these mechanisms is assigned an *Authentication LOA*, but with one proviso. No Authentication LOA can be higher than the Registration LOA, since it is the latter that originally authenticated who the user was. When a user logs in for a session, they are assigned a *Session LOA* that equates to the Authentication LOA of the authentication mechanism they chose to use.

Returning now to the linking service, we have made provisions to include the LOA in our protocol messages. When the linking service redirects the user to an IdP during the link registration phase, the user authenticates to the IdP with their preferred authentication mechanism, and this has an associated Authentication LOA. The IdP may return this as the current Session LOA to the linking service, along with the permanent identifier. The linking service stores this Session LOA as the Registration LOA of the user for this permanent identifier/IdP tuple, as shown in Table 1.

Local User ID	**PID**	**IdP**	**Registration LOA**
Fred	A=12345	airmiles.com	1
Fred	EduPersonID=u23@kent.ac.uk	kent.ac.uk	2
Fred	PID=4567890	XYX.co.uk	1
Fred	UID=qwertyuiop	cardbank.com	3
Etc……			

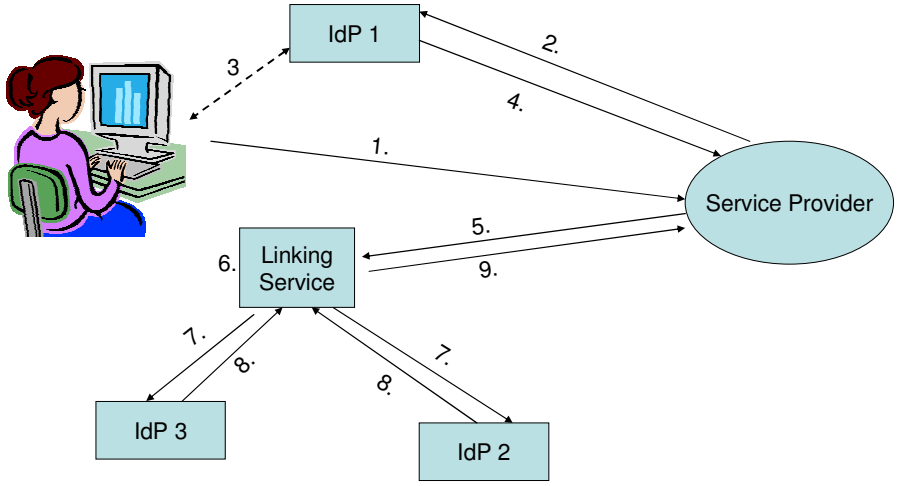
TABLE 1. An Example IdP Linking Table

Service Provision Phase

When a user wishes to use a web service, she first contacts the web site. The SP does not know who the user is, so must therefore redirect her to her IdP for authentication. In the Shibboleth model, the SP does this either directly (in a small federation) or indirectly via a Where Are You From service. In CardSpace, the SP returns the user to her Identity Selector whereupon she picks a card which represents her chosen IdP. She then presents her authentication credentials to the IdP, either directly in Shibboleth, or indirectly in CardSpace. The authentication dialogue is enhanced when attribute aggregation is supported, by asking the user if she wishes to use attribute aggregation with this SP. This can simply be a tick box alongside the username/password screen.

With direct IdP authentication, the IdP is able to show this enhanced screen since it knows if it has already generated one or more permanent identifiers for this user with one or more linking services. With CardSpace, the CardSpace application is able to show this enhanced screen if the SP says that it supports attribute aggregation (but in this case CardSpace does not actually know if the user has already set up links or not. This could be achieved by the IdP issuing a new card to the Identity Selector after it has established a permanent identifier for this user with a linking service.)

If the user chooses to perform attribute aggregation, the IdP includes one or more *referrals* in its response to the SP. A referral in effect says "you may find additional attributes for this user at this provider". A referral in this instance points to a linking service, and includes the permanent identifier of the user encrypted to the public key of the linking service. When the SP receives the authentication assertion containing the user's identity attributes, if these are sufficient for the requested service, then access will be granted. If they are not sufficient, and the SP supports attribute aggregation, it will follow the referral(s) by forwarding it(them) to the linking service(s) along with the authentication assertion (to prove that the user has been authenticated). It sets a Boolean in the request either asking the linking service to perform the aggregation, or saying it will perform the aggregation itself.



1. User makes a service request. **2.** User is redirected to her chosen IdP **3.** User authenticates to IdP 1. **4.** IdP 1 returns an authentication statement + attribute assertions + referral to linking service **5.** SP follows referral **6.** Linking service looks up IDP 1:PID 1 of user and finds links to other IdPs. **7.** Linking service requests attributes from linked IdPs using respective PIDs **8.** IdPs return signed and encrypted (to SP) attribute assertions. **9.** Linking service relays all attribute assertions to SP.

Figure 2. Attribute Aggregation Performed by Linking Service

When the linking service receives the referral, it decrypts the permanent identifier and searches for this in its IdP linking table. Once it has located the appropriate table entry, it retrieves the other table entries for the same user. Next it looks in its link release policy table to see which of the linked IdPs can be sent to this SP. If the SP asked to perform the aggregation, then referrals to the allowed IdPs are returned, with the permanent identifiers encrypted to their respective IdPs. The SP then follows these referrals in the same way that it did with the original one(s). If the SP asked the linking service to perform aggregation on its behalf, the linking service sends attribute query requests to the allowed IdPs, forwarding the name of the SP and the authentication assertion, so that the IdPs can encrypt their responses to the public key of the SP and tie the attributes to the

identifier found in the authentication assertion. Finally the IdPs digitally sign their responses. In this way the SP ultimately receives an authentication assertion and multiple encrypted attribute assertions, all digitally signed by their authoritative sources, and all containing the same random user identifier as in the authentication assertion. Since the SP trusts all these authoritative sources it can be assured that the user possesses all of the encapsulated identity attributes, even though the identifier of the user is random. The SP can make its access control decision based on the user's attributes and not on the identifier.

Using the LOA in Service Provision

The linking service may have stored Registration LOAs in its IdP Linking Table during the user's link registration phase. Though not essential, they serve to improve the performance of all subsequent user-SP sessions. During a user's service session, the linking service will only utilise linked IdPs whose Registration LOAs are higher than or equal to the current Session LOA provided by the IdP which authenticated this user's session. This prevents a user from creating links with low levels of assurance, and subsequently using them at higher Session LOAs, thereby pretending that the attributes have a high level of assurance. Conversely, a user is allowed to create links at high Registration LOAs, and then subsequently use them on lower Session LOAs, since the SP will know that the attributes can only be trusted up to the level of the current Session LOA.

If the linking service has stored the user's Registration LOA for a linked IdP, and a subsequent user-service session is authenticated by a different IdP at a lower Session LOA than this, the linking service is allowed to create a referral to the linked IdP. The linked IdP may then return user attributes at this low Session LOA. If however the subsequent user-session is authenticated at a higher Session LOA than the Registration LOA, the linking service should not create a referral to the linked IdP, since the linked IdP should always refuse to return any attributes for the user in this high Session LOA. This is because its attributes have not been assured to such a high level and it breaks the original proviso that no Authentication LOA can be higher than the original Registration LOA.

If the linking service has not stored the user's Registration LOA for a linked IdP, then the linking service will need to create a referral to this linked IdP for all subsequent user-service sessions, providing it is allowed by the link release policy, and the IdP will need to follow the same rules as above when deciding if the Session LOA is low enough to allow the user's attributes to be returned.

Link Release Policy

Comment [DWC2]: Can be a side bar

The user is allowed to create an IdP link release policy. This tells the linking service which linked IdPs should be released to which SPs. In the simplest case, the user might indicate that all linked IdPs can be released to all SPs. This will normally be the default policy (and would be indicated by an * in each of the columns of a link release policy table). In the most complex case, the user will require a different set of linked IdPs to be used with each SP. An example of such a policy for the user known locally to the linking

service as Fred is shown in table 2. This policy indicates that books.co.uk can receive attributes from three IdPs (airmiles.com, kent.ac.uk and cardbank.com); cardbank.com can receive attributes from all linked IdPs; and any other SP should only receive attributes from kent.ac.uk. The reason that both the permanent identifier and IdP are held in this table is because the user may have two different identities with one IdP, and might wish to link these together in a SP session.

Local User ID	SP	PID	IdP
Fred	books.co.uk	A=12345	airmiles.com
Fred	cardbank.com	*	*
Fred	books.co.uk	EduPersonID=u23@kent.ac.uk	kent.ac.uk
Fred	books.co.uk	UID=qwertyuiop	cardbank.com
Fred	*	EduPersonID=u23@kent.ac.uk	kent.ac.uk
Etc…			

TABLE 2. IdP Link Release Policy Table

Mapping to Standard Protocols

Our conceptual model has been mapped to the Security Assertions Markup Language (SAML) v2 protocol [9] during the link registration phase, and to both Liberty Alliance and CardSpace protocols during the service provision phase. Our attribute aggregation model provides for the passing of the LOA between the various components, but this is currently not part of the SAMLv2 specification. However OASIS is currently working on a SAML profile of the NIST LOA recommendation, and this is what we have used.

Link Registration Protocol

The link registration protocol uses standard SAML v2.0 authentication request/response messages [9] to request user authentication by a selected IdP and return a persistent identifier to the linking service. Upon receipt of the permanent identifier in the SAML response, the linking service will either find an existing entry in the IdP Linking Table for this permanent identifier/IdP tuple, or a new entry will be created in the table. Either way, the user can then link additional IdP accounts to this one.

In order to ensure that the IdP always returns a persistent identifier to the linking service, the SAML authentication request is constrained in the following ways:

- the Format attribute of the <NameIdPolicy> element is set to“urn:oasis:names:tc:SAML:2.0:nameid-format:persistent”
- the allowCreate attribute of the <NameIdPolicy> element is set to true, which allows the IdP to create a persistent identifier the first time around.

Service Provision Protocols

We devised two possible protocol mappings for attribute aggregation using Liberty Alliance protocols, and one using CardSpace protocols. All three mappings encode referrals as Liberty Alliance ID-WSF Endpoint References (EPRs) according to the EPR generation rules defined in Section 4.2 of [10]. Each EPR points to a linking service or IdP where the SP can find additional attributes for the user and the <sec:Token> of each EPR contains the encrypted permanent identifier of the user at the IdP. The Liberty

Alliance mapping we have implemented uses the Liberty ID-WSF Discovery Service [11].

Service Provision using Liberty Alliance Discovery Service

After the user contacts the SP, the SP issues a SAML authentication request message which the user's browser passes to the IdP. This message asks the IdP to generate a random identifier for the user in the authentication response (by setting the Format attribute to "urn:oasis:names:tc:SAML:2.0:nameid-format:transient") and to return both attributes and referrals (EPRs) in the response. The SAML response consists of a single SAML assertion which contains a single SSO assertion containing three statements: an SSO authentication statement, an attribute statement containing the users attributes and an attribute statement containing the EPR(s) of the linking service(s). The authentication statement contains the Session LOA.

Once the SP has received the SAML response it may attempt to access each of the referral EPR's using the discovery service. The Liberty Alliance IDWSF DiscoveryQuery operation (Section 3.3 of [11]) enables an IDWSF discovery service to be queried for relevant endpoint references that can be used to access other web based services. The DiscoveryQuery operation consists of a Query message and a QueryResponse. We define two types of DiscoveryQuery service. The first type, sent from an SP to a linking service, asks for the Discovery Services of the linked IdPs. The second type, sent from a linking service or SP to an IdP or linking service, asks for the EPR of the recipient's SAML v2.0 Attribute Authority so that it can subsequently be queried for the user's attributes. The IdP is able to map the permanent identifier in the DiscoveryQuery message to the local identifier of the user before the attribute query is sent to the AA.

If we left the design exactly like this, it would mean that the SP would need to create two different types of discovery query message depending upon whether it was talking to an IdP or linking service. In order to make the design more flexible, and to allow implementers to replace IdPs with linking services recursively, we have designed the protocol so that a single DiscoveryQuery message contains a request for both types of service and is sent to both types of recipient. The recipient knows what type of service it can provide, so it knows how to respond to the query, and which of the ServiceType elements inside the Query to ignore. Consequently only one service is returned in the response to each request. This means that the SP does not need to know whether it is talking to a linking service or an IdP and can create its DiscoveryQuery messages in exactly the same way to both.

The DiscoveryQuery message contains the <sec:Token> copied from the referral EPR and the initial authentication assertion in the message's SOAP header. This is the only non standard part of the protocol. In the original SOAP binding only the <sec:Token> would be present. The linking service decrypts the permanent identifier, retrieves the linked IdPs, and extracts the Session LOA from the authentication assertion. If the SP is performing aggregation, the linking service returns an ID-WSF QueryResponse message containing referral EPRs to the discovery services of the user's linked IdPs that have Registration LOAs greater or equal to the Session LOA. The SP then sends a

DiscoveryQuery message to each IdP's discovery service, requesting the EPR of the AA of the user. Alternatively, if the linking service is performing the aggregation, it sends the same message to each IdP. The IdP's discovery service locates the user's account by decrypting the permanent identifier, and if the user's Registration LOA is greater or equal to the presented Session LOA it maps the random identifier from the authentication assertion into the user's account. The IdP returns a QueryResponse containing either the EPR of the AA where the random identifier is now valid, or null if the query was invalid e.g. the Session LOA was too high. The SP (or linking service) sends a standard <samlp:AttributeQuery> to the AA, using the random identifier, whereupon the AA returns a standard <samlp:Response>, encrypted so that only the SP can retrieve the attributes.

Service Provision with the CardSpace Protocols

We have devised a protocol mapping for performing attribute aggregation within the Microsoft CardSpace infrastructure, which requires only minor changes to the CardSpace Identity Selector client and to the WS-Trust protocol.

After the user contacts the SP and is referred back to his CardSpace identity selector, the latter picks up the SP's security policy using the WS Metadata exchange protocol. The user picks a card and enters his login details to the prompt. If the SP has stated in its service metadata that it can accept referral attributes a check box labelled "do you want to use your linked cards in this transaction?" appears below the authentication dialog. If clicked, CardSpace attempts to get the user's claims using a modified WS-Trust message which contains a new Boolean attribute "aggregateIdentities" which states that referrals should be returned along with the user's attributes. Assuming the user's authentication credentials are correct, the IdP returns a CardSpace "request security token response" message which contains a single SAML SSO assertion containing three statements; an authentication statement, a SAML attribute statement containing the user's attributes, and if the user has linked this IdP to one or more linking services an additional SAML attribute statement containing referral(s) to the linking service(s). CardSpace relays this assertion to the SP which utilizes these referrals to perform attribute aggregation using the Liberty Alliance discovery protocol described above.

Conclusions

Federated identity management systems are now starting to be rolled out in significant numbers. But most have one severe limitation, which is that only one of the user's IdPs can be chosen per service session. To counteract this deficiency, we have designed and built a linking service which allows a user to link his various IdP accounts together in a privacy preserving manner. These linked accounts can then be used automatically during service provision to provide the aggregation of attributes from multiple authoritative sources, without necessitating the user to authenticate separately to each IdP. The system is based on limited trust between the IdPs, SPs and the linking service. We have mapped our conceptual model onto existing standard protocols based on SAML, Liberty Alliance and CardSpace, and have implemented the SAML and Liberty Alliance specifications.

This will be released as open source software as part of the PERMIS software suite[1]. We propose to implement the CardSpace protocols next.

Acknowledgements

The authors would like to thank the UK JISC and EC FP7 for funding this research under the Shintau and TAS3 projects respectively.

References

[1] 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

[2] Johnston, W., Mudumbai, S., Thompson, M. "Authorization and Attribute Certificates for Widely Distributed Access Control," IEEE 7th Int Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), Stanford, CA. June, 1998. pp 340 -345

[3] Liberty Alliance. "ID-FF 1.2 Specifications" Available from http://www.projectliberty.org/liberty/resource_center/specifications/liberty_alliance_id_ff_1_2_specifications

[4] David Chadwick. "Authorisation using Attributes from Multiple Authorities" in Proceedings of WETICE 2006, June 2006, Manchester, UK.

[5] Jill Gemmill, John-Paul Robinson, Tom Scavo, Purushotham Bangalore "Cross-domain authorization for federated virtual organizations using the myVocs collaboration environment" Concurrency and Computation: Practice and Experience Published online July 2008

[6] N. Klingenstein. "Attribute Aggregation and Federated Identity," International Symposium on Applications and the Internet Workshops (SAINTW'07), 2007, pp.26

[7] David Chadwick, George Inman, Nate Klingenstein. "Authorisation using Attributes from Multiple Authorities – A Study of Requirements". HCSIT Summit - ePortfolio International Conference, October 2007, Maastricht.

[8] William E. Burr, Donna F. Dodson, Ray A. Perlner, W. Timothy Polk, Sarbari Gupta, Emad A. Nabbus. "Electronic Authentication Guideline", NIST Special Publication NIST Special Publication 800-63-1, Feb 2008

[9] OASIS. "Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0", OASIS Standard, 15 March 2005

[10] Hodges, J. Aarts, R. Madsen, P. and Cantor, S.(Editors). "Liberty ID-WSF Authentication, Single Sign-On, and Identity Mapping Services Specification v2.0". Liberty Alliance Project.

[11] Hodges, J. Cahill, C.(Editors). "Liberty ID-WSF Discovery Service Specification V2.0". Liberty Alliance Project

[1] See www.openpermis.org and sec.cs.kent.ac.uk/permis