Attribute Aggregation Protocols

Authors: George Inman, David Chadwick. University of Kent

Release Number	Date	Author
1.0	20th November 2007	G.Inman
1.1	5th December 2007	G. Inman
1.2	18th Dec 2007	D. Chadwick
1.3	22nd Dec 2007	G.Inman
1.4	20th Jan 2007	G.Inman
1.5	4th Feb 2007	G.Inman
1.6	16 Mar. 08	D. Chadwick
1.6.1	11th April 08	G. Inman
1.7	21st April 08	G.Inman
1.8	20th September 08	G.Inman
1.8.1	22nd December 08	G.Inman

Contents

1. Introduction

Whilst there are several virtual organisation (VO) solutions currently in production (e.g. Shibboleth [5], Globus Toolkit [8], CardSpace [11]) they have all to some extent been hampered by the same problem, which is a lack of a standards based approach to the problem of aggregating attributes from multiple identity providers (IdPs) for use by a single service provider (SP) in an access control decision. Several ad-hoc solutions to this problem have been experimented with, such as GridShib [1] and My-Vocs [2]. Whilst at first glance these may appear to offer elegant solutions to the problem space, on deeper inspection their models are usually flawed. For example the MyVocs model requires that an SP trusts the MyVocs server both to authenticate all users and to pass attribute assertions signed by the MyVocs server rather than by their original IdP sources. The GridShib solution as originally conceived could only cater for one other Shibboleth IDP and all linked users have to have their permanent IDs hard wired together.

In order to develop a standards based solution to the problem of attribute aggregation, we have embarked upon a two year project to define a set of standardised protocols that can aggregate attributes from any number of IdPs, whilst maintaining user privacy and satisfying the majority of user requirements as collected by our recent user survey [3]. User requirements were elicited through the wide circulation of a structured questionnaire, and the analysis of the results is presented in [3]. We then defined a conceptual model [4] that satisfies the majority of the user requirements. The conceptual model employs a new component called a Linking Service (LS), which is a trusted third party under the control of the user, whose purpose is to link together the different IdPs that hold a user's attributes. The linking is done in a privacy preserving way by using random (but unique and permanent) identifiers generated by the IdPs, so that the LS is unaware of the true identity of the user. The protocol in the conceptual model has a referral, which is a pointer from one IdP to the LS or from the LS to an IdP, which serves to link the authentication handle (used to identify the user in the current session) with the random permanent identifier used by the IdP to identify the user. The conceptual model has a number of different modes of protocol interaction between the IdP, LS and SP, in which attribute aggregation can be undertaken by either the LS or the SP. The next task in the project is to map the conceptual protocol message flows onto one or more standard protocols, prior to implementation.

This document describes four interchangeable standardised protocol message flows for communication between the IdPs, SP and LS, each with their own strengths and weaknesses. The authors would like to obtain feedback from the community about these protocol options before implementation begins.

The rest of this document is structured as follows. Section 2 describes the linking protocol and Section 3 describes each of the protocol's message flows. Section 4 evaluates each protocol's conformance to existing standards. Section 5 looks at the changes that would be needed to the existing Shibboleth infrastructure if each protocol was to be implemented. Other systems such as Liberty Alliance or CardSpace may require different changes in order to conform to the suggested protocol flows. Section 6 evaluates each protocol's support of the conceptual model set out in [4]. Section 7 looks at how each protocol supports the initial requirements set out in [3]. Finally Section 8 provides a summary of the pros and cons of each protocol and presents our conclusions.

2. IdP Linking Protocol Description

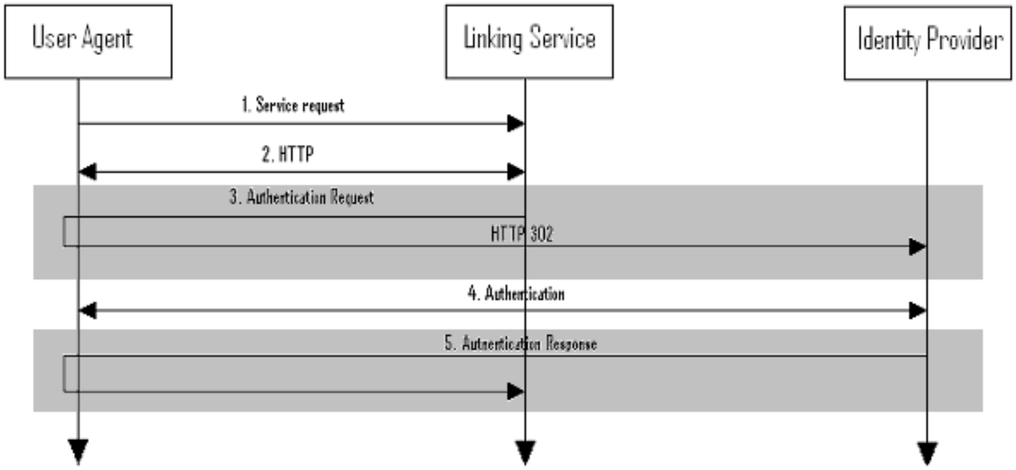
Consider the use case where a principal wishes to link an existing account at an identity provider to a new account at the linking service. This is accomplished by the user contacting the linking service, then selecting his or her identity provider from a protocol exchange with the linking service, after which the user is redirected to the IdP for authentication. The identifier, in the resulting authentication assertion returned by the IdP to the linking service, is then used to create a database entry at the linking service for the user.

This use case is based upon the following assumptions

- The principal possesses an account at the Identity Provider
- The principal has a client that issues a request to link an Identity Provider to a linking service
- The linking service has a pre-existing trust relationship with the principal's Identity Provider and is able to query that identity provider for authentication information about the principal
- The Identity Provider must be capable of holding a persistent unique identifier for each principal and must be prepared to return this to the linking service.
- The linking service is able to store multiple identity providers' entityIDs against a locally generated principal name in such a way that they can be linked to a single principal in its security domain.

The sequence of steps for the full use case is shown below.

> **Note:** a grey box highlights the steps constrained by this profile. The other steps are shown only for completeness; the profile does not constrain them.



1. **HTTP request to Linking Service**

   In step 1, the principal, via an HTTP user agent, makes an HTTP request to link an Identity provider to the linking service, without a security context

2. **HTTP exchange to determine Identity Provider to be linked**

   In step 2, the linking service determines which Identity provider the principal wishes to link to by some means outside the scope of this specification. One example would be for the linking service to send a list of known IdPs to the user agent, from which one can be selected.

3. **Authentication Request issued by Linking service to Identity Provider**

In step 3, the linking service issues a <samlp:AuthnRequest> message and redirects the user agent to the identity provider.

The <samlp:AuthnRequest> is created according to the rules below:

- The requests optional <saml:Subject> element SHOULD not be defined.
- The NameIDPolicy element of the request SHOULD be defined as follows:
  - The Format attribute MUST be set to "urn:oasis:names:tc:SAML:2.0:nameid-format:persistent". Thereby specifying that the queried IdP MUST return a permanent persistent identifier valid between the LS and the queried IdP
  - The SPNameQualifier attribute MUST be set to the LS.
  - The AllowCreate attribute SHOULD be set to true, thereby allowing the IdP to create a new persistent identifier.
- The saml:Conditions element MAY contain NotBefore or NotOnOrAfter attributes if the SP wishes to limit the validity of the response.
- The RequestedAuthnContext attribute MAY be specified if the SP requires a specific authentication method to be used but its use is not required.
- The Scoping element of the request MAY be used to specify the RequesterID if the query is to be proxied by the requested IdP to another IdP to perform authentication.
- The ForceAuthn attribute MUST be true requiring that the principal be freshly authenticatated rather than allowing the use of pre-existing authentication data.
- The IsPassive attribute MUST not be defined
- The AssertionConsumerServiceURL attribute SHOULD set to the issuer of the request thereby specifying the ultimate consumer of the response.
- The AssertionConsumerServiceIndex attribute SHOULD not be specified
- The ProtocolBinding attribute is not specified in this document but MAY be included in the request.
- The AttributeConsumingServiceIndex attribute MUST not be set.
- The ProviderName attribute MAY contain the human readable name of the requester.

e.g.

```
<samlp:AuthnRequest xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
ID="Request1" Version="2.0"  IssueInstant="2007-11-26T07:35:00Z " ForceAuthn="true"
AssertionConsumerServiceURL="ls.kent.ac.uk"  ProviderName="Kent SP" >
    <saml:NameIDPolicy Format=" urn:oasis:names:tc:SAML:2.0:nameid-format:persistent"
    SPNameQualifier="sp.kent.ac.uk" AllowCreate="true" />
    <saml:Conditions NotBefore=""2007-11-26T07:35:00Z"  NotOnOrAfter="2007-11-
26T07:40:00Z"  />
</samlp:AuthnRequest>
```

4. Identity Provider identifies Principal

In step 4, the principal is identified by the identity provider by some means outside the scope of this specification. This may require a new act of authentication, or it may reuse an existing authenticated session.

5. **Identity Provider issues <samlp:Response> to Linking Service**

   In step 5, the identity provider issues a SAML <samlp:Response> message to be delivered by the user agent to the linking service. The Linking service then uses the persistent name identifier within the <subject> element of the <AuthnStatement> assertion to create a link against a randomly generated user identifier in its data store.

   The <samlp:Response> message issued by the identity provider MAY optionally contain an <AuthnContext> element specifying the method of authentication which can then be used to represent the level of authentication at which the user authenticated at, to be stored in the LS’s database.

   e.g.

```
<samlp="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
ID="Response1 "
InResponseTo="Request1" Version="2.0"
IssueInstant="2007-11-26T07:35:00Z" Destination="http://ls.kent.ac.uk/"
Consent="urn:oasis:names:tc:SAML:2.0:consent:obtained”>

  <saml2: Assertion Version="2.0" IssueInstant="2007-11-26T07:35:00Z" ID="authn1">
       <saml2:Issuer>https://idp1.kent.ac.uk</saml2:Issuer>
       <ds:Signature>...</ds:Signature>
       <saml2:Subject>
              <saml2:NameID>
                     persistentID1
              </saml2:NameID>
       </saml2:Subject>
       <saml2:AuthnStatement AuthnInstant=”2007-11-26T07:35:00Z”
       SessionNotOnOrAfter =”2007-11-26T07:40:00Z” >
              <saml2:AuthnContext>
                     <AuthnContextClassRef>
                     urn:mace:dir:constant:nist-sp-800-63:1
                     </AuthnContextClassRef>
              </saml2:AuthnContext>
              <saml2:SubjectLocality  Address=”129.12.16.129”
              DNSName=”https://idp1.kent.ac.uk” />
       </saml2:AuthnStatement>
       <saml2:Conditions NotOnOrAfter="2007-11-26T07:40:00Z ">
              <saml2:AudienceRestriction>
                     <saml2:Audience>ls.kent.ac.uk</saml:Audience>
              </saml2:AudienceRestriction>
       </saml2:Conditions>
</saml2: Assertion>
```

2.1 Linking additional identity providers to a pre-existing account at the linking service

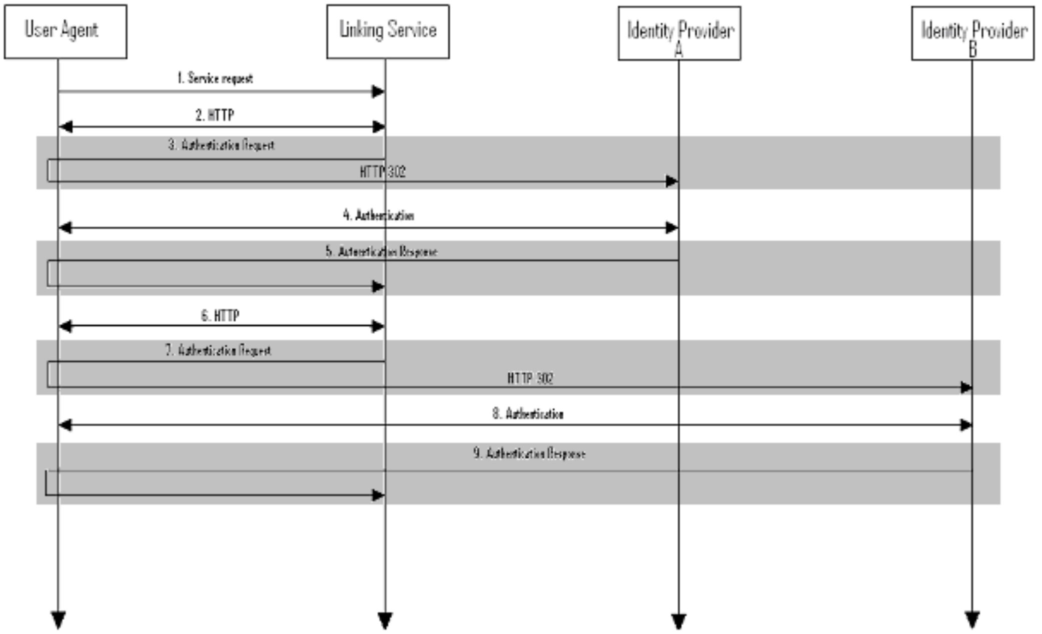
Now consider the case where the principal wishes to link additional identity providers to an existing account on the linking service. He is first asked to authenticate to an identity provider for which the linking service already holds an entry and the name identifier contained in the authentication response is used to identify the existing account at the linking service. Once this account has been identified the Linking service asks the principal to authenticate to the new identity provider and the name identifier contained in the second authentication response is then linked to the existing identifier at the linking service.

This use case is based upon the following assumptions

- The principal posses an account at both Identity Providers A and B
- The principal has a client that can issue a request to link an Identity Provider to an existing account at a linking service
- The Identity Provider must be capable of holding a persistent unique identifier for each principal and must be prepared to return this to the linking service.
- The linking service has pre-existing trust relationships with both Identity Providers A and B and is able to query both identity providers for authentication information
- The linking service is able to store multiple identity providers' entityIDs against a locally generated principal name in such a way that they can be linked to a single principal in its security domain.

The sequence of steps for the full use case is shown below.

> **Note:** a grey box highlights the steps constrained by this profile. The other steps are shown only for completeness; the profile does not constrain them.



1. **HTTP request to Linking Service**

   In step 1, the principal, via an HTTP user agent, makes an HTTP request to link a new Identity provider to a pre-existing account at the linking service, without a security context

2. **HTTP exchange to determine what identity provider/s the principal has already linked**

   In step 2, the linking service determines which Identity provider the principal has already linked to by some means outside the scope of this specification. For example, the linking service could send a list of known IdPs to the user agent, from which the user can select one that he/she has already linked to (IdP A).

3. **Authentication Request issued by Linking service to Identity Provider**

   In step 3, the linking service issues an <samlp:AuthnRequest> to the IdP and redirects the user agent to identity provider A.

The <samlp:AuthnRequest> is created according to the rules below:

- The requests optional <saml:Subject> element SHOULD not be defined.
- The NameIDPolicy element of the request SHOULD be defined as follows:
  - The Format attribute MUST be set to "urn:oasis:names:tc:SAML:2.0:nameid-format:persistent". Thereby specifying that the queried IdP MUST return a permanent persistent identifier valid between the LS and the queried IdP

  - The SPNameQualifier attribute MUST be set to the LS.
  - The AllowCreate attribute SHOULD be set to false, forcing the IdP to return a pre-existing identitfier for the principal valid between the LS and the queried IdP.
- The saml:Conditions element MAY contain NotBefore or NotOnOrAfter attributes if the SP wishes to limit the validity of the response.
- The RequestedAuthnContext attribute MAY be specified if the SP requires a specific authentication method to be used but its use is not required.
- The Scoping element of the request MAY be used to specify the RequesterID if the query is to be proxied by the requested IdP to another IdP to perform authentication.
- The ForceAuthn attribute MUST be true requiring that the principal be freshly authenticatated rather than allowing the use of pre-existing authentication data.
- The IsPassive attribute MUST not be defined
- The AssertionConsumerServiceURL attribute SHOULD be set to the issuer of the request, thereby specifying the ultimate consumer of the response.
- The AssertionConsumerServiceIndex attribute SHOULD not be specified
- The ProtocolBinding attribute is not specified in this document but MAY be included in the request.
- The AttributeConsumingServiceIndex attribute MUST not be set.
- The ProviderName attribute MAY contain the human readable name of the requester.

e.g.

```
<samlp:AuthnRequest xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
ID="Request1" Version="2.0"  IssueInstant="2007-11-26T07:35:00Z " ForceAuthn=”true”
AssertionConsumerServiceURL=”ls.kent.ac.uk”  ProviderName=”Kent SP” >
<saml:NameIDPolicy Format=” urn:oasis:names:tc:SAML:2.0:nameid-format:persistent”
SPNameQualifier=”sp.kent.ac.uk” AllowCreate=”true” />
        <saml:Conditions NotBefore=“"2007-11-26T07:35:00Z"  NotOnOrAfter="2007-11-
        26T07:40:00Z"  />
</samlp:AuthnRequest>
```

4. Identity Provider identifies Principal

In step 4, the principal is identified by identity provider A by some means outside the scope of this specification. This may require a new act of authentication, or it may reuse an existing authenticated session.

5. Identity Provider issues <samlp:Response> to Linking Service

In step 5, the identity provider issues a SAML <samlp:Response> message to be delivered by the user agent to the linking service. The Linking service then uses the name identifier within the <subject> element of the <AuthnStatement> assertion to map the user to a pre-existing account at the linking service.

e.g.

```
<samlp="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
ID="Response1 "
InResponseTo="Request1" Version="2.0"
IssueInstant="2007-11-26T07:35:00Z" Destination="http://ls.kent.ac.uk/"
Consent="urn:oasis:names:tc:SAML:2.0:consent:obtained”>

  <saml2: Assertion Version="2.0" IssueInstant="2007-11-26T07:35:00Z" ID="authn1">
        <saml2:Issuer>https://idp1.kent.ac.uk</saml2:Issuer>
        <ds:Signature>...</ds:Signature>
        <saml2:Subject>
                <saml2:NameID>
                        persistentID1
                </saml2:NameID>
        </saml2:Subject>
        <saml2:AuthnStatement AuthnInstant=”2007-11-26T07:35:00Z”
        SessionNotOnOrAfter =”2007-11-26T07:40:00Z” >
                <saml2:AuthnContext>
                        <AuthnContextClassRef>
                        urn:mace:dir:constant:nist-sp-800-63:1
                        </AuthnContextClassRef>
                </saml2:AuthnContext>
                <saml2:SubjectLocality  Address=”129.12.16.129”
                DNSName=”https://idp1.kent.ac.uk” />
        </saml2:AuthnStatement>
        <saml2:Conditions NotOnOrAfter="2007-11-26T07:40:00Z ">
                <saml2:AudienceRestriction>
                        <saml2:Audience>ls.kent.ac.uk</saml:Audience>
                </saml2:AudienceRestriction>
        </saml2:Conditions>
</saml2: Assertion>
```

Note. If no pre-existing account can be found for the principal at the linking service the linking service MAY ask if the principal wishes to create an account using the authentication details provided from IdP A as in the message flow described in Section 2.1.1.

6. HTTP exchange to determine what identity provider the principal wishes to link

In step 6 the linking service determines which Identity provider the principal wishes to link to the account identified in step 5, by some means outside the scope of this specification. For example, a list of known identity providers may be transferred to the user agent. The user chooses Idp B.

7. Authentication Request issued by Linking service to Identity Provider

In step 7, the linking service issues a <saml:AuthnRequest> and redirects the user agent to identity provider B.

The <samlp:AuthnRequest> is created according to the rules below:

- The requests optional <saml:Subject> element SHOULD not be defined.
- The NameIDPolicy element of the request SHOULD be defined as follows:
    - The Format attribute MUST be set to “urn:oasis:names:tc:SAML:2.0:nameid-format:persistent”. Thereby specifying that the queried IdP MUST return a permanent persistent identifier valid between the LS and the queried IdP
    - The SPNameQualifier attribute MUST be set to the LS.
    - The AllowCreate attribute SHOULD be set to true, thereby allowing the IdP to create a new persistent identifier.
- The saml:Conditions element MAY contain NotBefore or NotOnOrAfter attributes if the SP wishes to limit the validity of the response.
- The RequestedAuthnContext attribute MAY be specified if the SP requires a specific authentication method to be used but its use is not required.
- The Scoping element of the request MAY be used to specify the RequesterID if the query is to be proxied by the requested IdP to another IdP to perform authentication.
- The ForceAuthn attribute MUST be true requiring that the principal be freshly authenticatated rather than allowing the use of pre-existing authentication data.
- The IsPassive attribute MUST not be defined
- The AssertionConsumerServiceURL attribute SHOULD be set to the issuer of the request thereby specifying the ultimate consumer of the response.
- The AssertionConsumerServiceIndex attribute SHOULD not be specified
- The ProtocolBinding attribute is not specified in this document but MAY be included in the request.
- The AttributeConsumingServiceIndex attribute MUST not be set.
- The ProviderName attribute MAY contain the human readable name of the requester.

e.g.

```
<samlp:AuthnRequest xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
ID="Request2" Version="2.0"  IssueInstant="2007-11-26T07:35:00Z " ForceAuthn=”true”
AssertionConsumerServiceURL=”ls.kent.ac.uk”  ProviderName=”Kent LS” >
    <saml:NameIDPolicy Format=” urn:oasis:names:tc:SAML:2.0:nameid-format:persistent”
    SPNameQualifier=”sp.kent.ac.uk” AllowCreate=”true” />
    <saml:Conditions NotBefore=“"2007-11-26T07:35:00Z"  NotOnOrAfter="2007-11-
26T07:40:00Z"  />
</samlp:AuthnRequest>
```

8. Identity Provider B identifies Principal

In step 8, the principal is identified by identity provider B using some means outside the scope of this specification. This may require a new act of authentication, or it may reuse an existing authenticated session.

9. **Identity Provider issues <samlp:Response> to Linking Service**

In step 9, the identity provider issues a SAML <samlp:Response> message to be delivered by the user agent to the linking service. The Linking service then uses the name identifier within the <subject> element of the <AuthnStatement> assertion to create a link against the pre-existing account discovered (or the new account created) in step 5.

The <samlp:Response> message issued by the identity provider MAY optionally contain an <AuthnContext> element specifying the method of authentication which can then be used to represent the level of authentication at which the user authenticated at, to be stored in the LS's database.

e.g.

```
<samlp="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
ID="Response2 "
InResponseTo="Request1" Version="2.0"
IssueInstant="2007-11-26T07:35:00Z" Destination="http://ls.kent.ac.uk/"
Consent="urn:oasis:names:tc:SAML:2.0:consent:obtained”>

  <saml2: Assertion Version="2.0" IssueInstant="2007-11-26T07:35:00Z" ID="authn2">
        <saml2:Issuer>https://idp2.kent.ac.uk</saml2:Issuer>
        <ds:Signature>...</ds:Signature>
        <saml2:Subject>
                <saml2:NameID>
                        persistentID2
                </saml2:NameID>
        </saml2:Subject>
        <saml2:AuthnStatement AuthnInstant=”2007-11-26T07:35:00Z”
        SessionNotOnOrAfter =”2007-11-26T07:40:00Z” >
                <saml2:AuthnContext>
                        <AuthnContextClassRef>
                        urn:mace:dir:constant:nist-sp-800-63:1
                        </AuthnContextClassRef>
                </saml2:AuthnContext>
                <saml2:SubjectLocality  Address=”129.12.16.129”
                DNSName=”https://idp2.kent.ac.uk” />
        </saml2:AuthnStatement>
        <saml2:Conditions NotOnOrAfter="2007-11-26T07:40:00Z ">
                <saml2:AudienceRestriction>
                        <saml2:Audience>ls.kent.ac.uk</saml:Audience>
                </saml2:AudienceRestriction>
        </saml2:Conditions>
</saml2: Assertion>
```

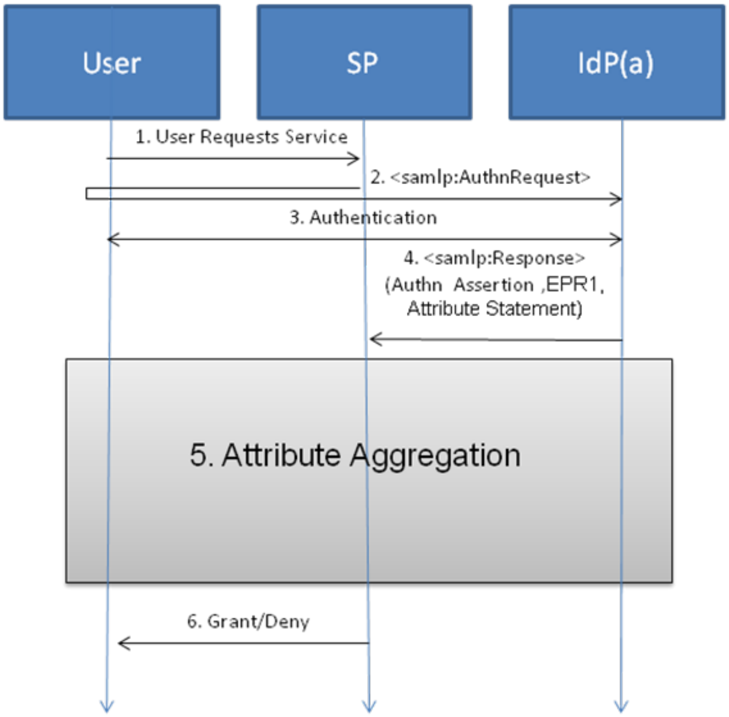
3. Aggregation Protocol Descriptions (Non Normative)

Each of the following protocol flows assumes that the user has two different sets of attributes held by IdP(a) (at https://idp1.kent.ac.uk) and IdP(b) (at https://idp2.kent.ac.uk), and that both of these IdPs have already been linked together at the linking service LS[1](at https://ls.kent.ac.uk). The user wishes to access a service at provider SP (at https://sp.kent.ac.uk), and needs to present both sets of attributes in order to be granted access.

3.1 Initial Authentication and Attribute Queries

For each of the aggregation protocol models described in this document both the act of authentication and the initial query for attributes from the authenticating IdP can be achieved using one of the two message flows and their related processing rules presented below.

3.1.1 IdP Direct Authentication



SP Authentication and Initial Attribute Query

1. The user requests a service at the service provider.
2. The SP sends a <samlp:AuthnRequest> protocol message via the users browser to IdP(a). How this is achieved is not specified. It could be either via a direct link to the trusted IdP(a) or via a WAYF service, or via the user choosing the IdP from his CardSpace desktop. This <samlp:AuthnRequest> includes a request for a set of attributes that have been predefined in the SP's metadata and this request is specified using the request's AttributeConsumingServiceIndex attribute. (see 4.1)
3. Authentication is performed at IdP(a) in the usual way except for one small change in that during authentication the user is also asked whether or not he

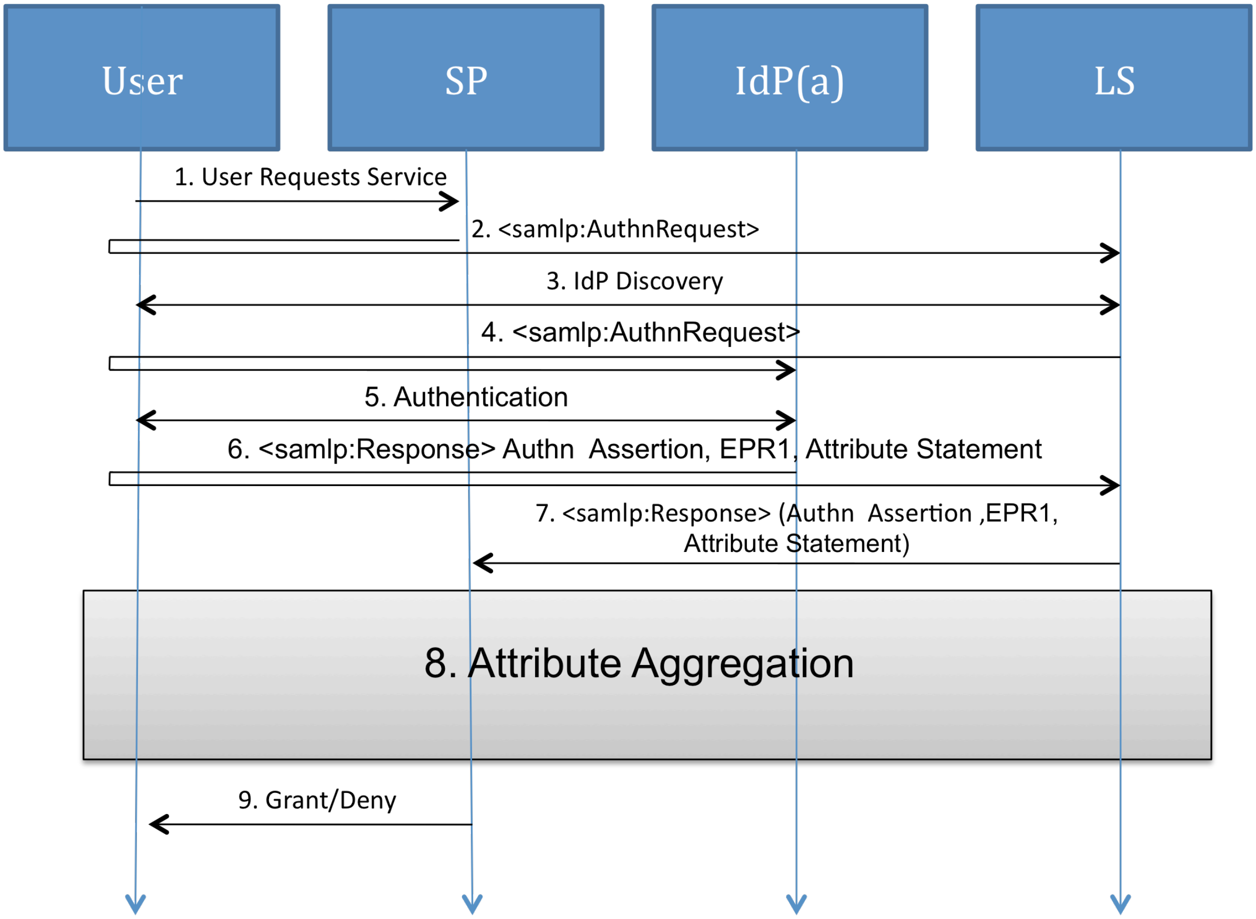
[1] Extension of each protocol flow to 3 or more linked IdPs should be intuitively obvious.

wishes to use to aggregation to return additional attributes, if so he is also asked which linked LS account(s) he would like to pull additional accounts and attributes from.

4. If the user chooses not to use aggregation then IdP(a) returns a <samlp:Response> containing a single SAML assertion which contains both the principals authentication and attribute statements to the SP.

   If however the principal wishes to use attribute aggregation then IdP(a) returns a new <Samlp:Response> message containing a single SAML assertion containing three SAML statements to the SP. The first of these statements is a new SSO authentication statement, which MAY contain a LOA attribute as the principals authentication context. The second statement contains the user's attributes stored at the IdP that match the attributes requested in the <samlp:AuthnRequest>. The third SAML statement should contain additional IDWSF Endpoint References (EPR) which specify how to access the linking service entities the principal wishes to use for aggregation and provide a security token for doing so (see 4.3). All of these EPR attributes are stored in a single attribute statement. The <saml:Subject> element of each this assertions should be a new random identifier that corresponds to the <saml:NameIDPolicy> element of the request. (see 4.2)

3.1.2 IdP via LS Authentication



IdP via LS Authentication and Initial Attribute Query

1. The user requests a service at the service provider.
2. The SP sends a <samlp:AuthnRequest> protocol message via the users browser to the LS. How this is achieved is not specified. It could be either via a direct link to the trusted IdP(a) or via a WAYF service, or via the user choosing the LS from his CardSpace desktop. This <samlp:AuthnRequest> includes a request for a set of attributes that have been predefined in the SP's metadata and this request is specified using the request's AttributeConsumingServiceIndex attribute. (see 4.1)
3. A HTTP exchange then takes place in order to determine which Identity Provider the user wishes to use for authentication.
4. Once the IdP to be used for authentication has been determined the LS sends a <samlp:AuthnRequest> protocol message via the users browser to the identity provider selected. This <samlp:AuthnRequest> is constructed according to the SAML ECP rules (Section * of [SAML-*]) and includes a request for the same set of attributes that were requested in the initial authentication request provided by the SP. (see 4.1)

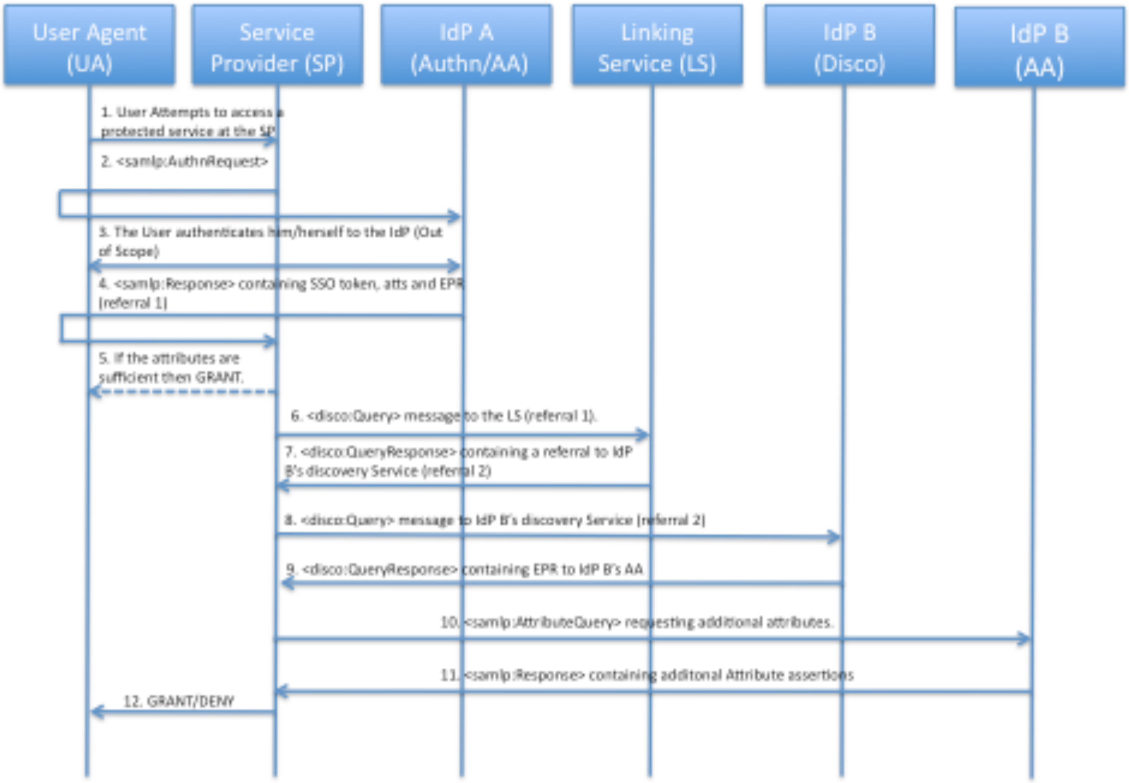
5. Authentication is performed at IdP(a) in the usual way except for one small change in that during authentication the user is also asked whether or not he wishes to use to aggregation to return additional attributes and if so he is also asked which linked LS account(s) he would like to pull additional accounts and attributes from
6. If the user chooses not to use aggregation then IdP(a) returns a <samlp:Response> containing a single SAML assertion which contains both the principals authentication and attribute statements to the LS.

   If however the principal wishes to use attribute aggregation then IdP(a) returns a new <Samlp:Response> message containing a single SAML assertion with three SAML statements to the SP. The first of these statements is a new SSO authentication statement, which MAY contain a LOA attribute as the principals authentication context. The second statement contains the user's attributes stored at the IdP that match the attributes requested in the <samlp:AuthnRequest>. The third SAML statement should contain additional IDWSF Endpoint References (EPR) which specify how to access the linking service entities the principal wishes to use for aggregation and provide a security token for doing so (see 4.3). All of these EPR attributes are stored in a single attribute statement. The SAML attribute statement containing the user's attributes contained within the response MUST be encrypted to the SP. The attribute assertion containing the EPR pointing to the LS within the response MAY also be encrypted to the LS. (see 4.2)

7. The LS processes the authentication response and maps the EPR contained in the attribute statement of the unencrypted assertion to an internal account identifier. The LS MAY then either perform SP based Aggregation or LS Based aggregation. If SP aggregation is to be used then a new <samlp:Response> is created. This <samlp:Response> message contains a new authentication assertion issued by the LS a new attribute statement containing EPR attributes and the original SSO assertion issued by IdP(a) containing the principals attributes. The new authentication assertion must conform to the Proxying Processing Rules for authentication assertions as set out in [6] section 3.4.1.5 namely that the relevant information is copied from the original assertion and that a new <saml:AuthenticatingAuthority> element specifying the IdP that performed the actual act of authentication is added to the assertion. The attribute assertion issued by the LS contains a new EPR attribute for each of the accounts stored in its database constructed using the referral createion rules set out in section 4.3. The soap message MUST also contain the attribute assertion containing the principals attributes at IdP(a). The <saml:Subject> element of each of these three assertions should be the random identifier that corresponds to the <saml:NameIDPolicy> element of the authentication response returned by IdP.

3.2 IdP Direct SP Aggregation

IdP direct SP Aggregation.


User Agent (UA)
Service Provider (SP)
IdP A (Authn/AA)
Linking Service (LS)
IdP B (Disco)
IdP B (AA)
1. User Attempts to access a protected service at the SP
2. <samlp:AuthnRequest>
3. The User authenticates him/herself to the IdP (Out of Scope)
4. <samlp:Response> containing SSO token, atts and EPR (referral 1)
5. If the attributes are sufficient then GRANT.
6. <disco:Query> message to the LS (referral 1).
7. <disco:QueryResponse> containing a referral to IdP B's discovery Service (referral 2)
8. <disco:Query> message to IdP B's discovery Service (referral 2)
9. <disco:QueryResponse> containing EPR to IdP B's AA
10. <samlp:AttributeQuery> requesting additional attributes.
11. <samlp:Response> containing additonal Attribute assertions
12. GRANT/DENY

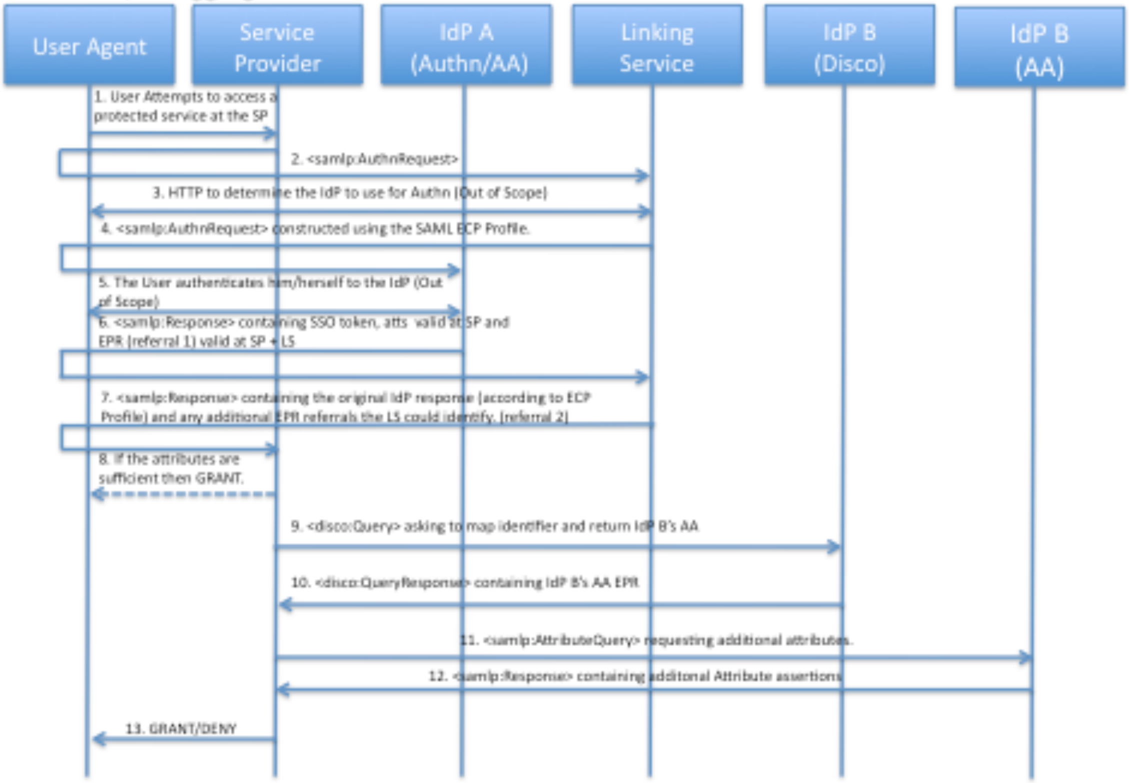
1. The user requests a service at the SP.
2. The SP sends a <samlp:AuthnRequest> protocol message via the users browser to IdP(a). (See Section 3.1.1 for additional information)
3. Authentication is performed at IdP(a) in the usual way except for one small change in that during authentication the user is also asked whether or not he wishes to return additional attributes and if so he is also asked which linked LS account(s) he would like to pull additional accounts and attributes from
4. If the user chooses not to use aggregation then IdP(a) returns a <samlp:Response> containing a single SAML assertion which contains both the principals authentication and attribute statements to the SP. If however the principal wishes to use attribute aggregation then IdP(a) returns a new <Samlp:Response> message containing three SAML statements to the SP as described in section 3.1.
5. The SP then process the authentication and attribute assertions. If the attributes returned are sufficient to authorise the principal then the process ends and the user is granted access to the resource.
6. If however insufficient attributes are returned to authorise the principal then the SP processes the third assertion containing additional identity account information and sends a new IDWSF discovery query message to the LS requesting additional user account information as EPRs. As in this protocol exchange the SP will aggregate additional attributes the <disco:Query> messages aggregate attribute should be set to false. (See 4.4)
7. The LS then processes the Discovery Query request using the encrypted persistent name identifier of the user contained in the security header to authenticate the

request and to identify an internal account. Once this has been accomplished the authentication assertion is examined to determine whether the service trusts the authenticating IdP and whether the authentication LoA is sufficient to release additional EPRs to the user. If so then the LS creates a new EPR for each linked identifier that matches the parameters of the request. These EPRs are then combined into a single IDWSF QueryResponse message and returns this message to the SP. The LS itself holds only links to the discovery services of the entities that have been linked to it so at this point only additional discovery service EPRs are returned. (See 4.5)

8. Any EPR's contained in the returned QueryResponse messages are then processed by the SP and the EPR service type is checked. If the endpoint reference refers to a discovery service then the discovery service is queried as before, if however the endpoint reference refers to a SAML2 Attribute Authority endpoint then a SAML2 <samlp:AttributeQuery> is generated using the initial random identifier as the request's subject. As the response has come from an LS entity any discovered EPRs will be for Discovery service entities and the Discovery Query message is constructed as in step 5.
9. The queried IdP then processes the discovery query request and maps the Subject of the Authentication statement issued by the authenticating IdP and contained in one of the request's <sec:Token> elements to the persistent identifier contained in the referral as in step 6. If the queried IdP trusts that the initial IdP has authenticated the user sufficiently then a DiscoveryResponse message is created containing a single EPR containing the address of the SAML 2.0 attribute authority at which the mapped random identifier is now valid.
10. The SP processes the returned messages as follows, Any EPR's contained in the returned QueryResponse messages are processed and the EPR service type is checked. If the endpoint reference refers to a discovery service then the discovery serice is queried as before, if however the endpoint reference refers to a SAML2 Attribute Authority endpoint then a SAML2 <samlp:AttributeRequest> is generated using the initial random identifier as the request's subject.
11. The IdP then processes the attribute query having mapped the random identifier used in the attribute request to the persistent identifier in step 9. A soap message is then created containing a <samlp:Response> message holding the principal's attributes. This <samlp:Response> is then signed and returned by the IdP to the LS. The subject of this <samlp:Response> is the random identifier generated in the original authentication response.
12. The SP processes each <samlp:Response> message decrypts the attributes in the returned assertions and uses the combined set of returned attributes for authorisation.

3.3 IdP via LS SP Aggregation

IdP via LS, SP Aggregation.


User Agent
Service Provider
IdP A (Authn/AA)
Linking Service
IdP B (Disco)
IdP B (AA)
1. User Attempts to access a protected service at the SP
2. <samlp:AuthnRequest>
3. HTTP to determine the IdP to use for Authn (Out of Scope)
4. <samlp:AuthnRequest> constructed using the SAML ECP Profile.
5. The User authenticates him/herself to the IdP (Out of Scope)
6. <samlp:Response> containing SSO token, atts valid at SP and EPR (referral 1) valid at SP + LS
7. <samlp:Response> containing the original IdP response (according to ECP Profile) and any additional EPR referrals the LS could identify. (referral 2)
8. If the attributes are sufficient then GRANT.
9. <disco:Query> asking to map identifier and return IdP B's AA
10. <disco:QueryResponse> containing IdP B's AA EPR
11. <samlp:AttributeQuery> requesting additional attributes.
12. <samlp:Response> containing additonal Attribute assertions
13. GRANT/DENY

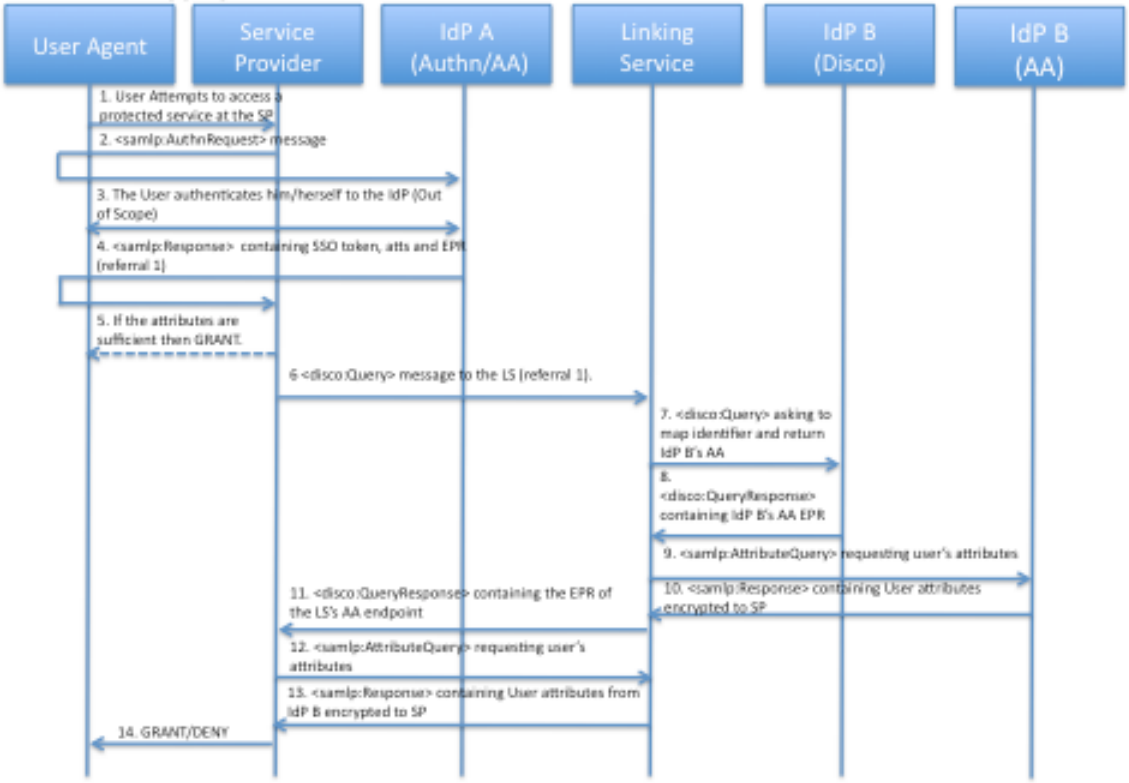
1. The user requests a service at the SP.
2. The SP sends a <samlp:AuthnRequest> protocol message via the users browser to the LS. (See Section 3.1.2 for additional information and processing rules)
3. The LS determines the IdP to use for authentication.
4. The LS sends a new <samlp:AuthnRequest> protocol message via the users browser to the LS. (See Section 3.1.2 for additional information and processing rules)
5. Authentication is performed at IdP(a) in the usual way except for one small change in that during authentication the user is also asked whether or not he wishes to use to aggregation to return additional attributes and if so which linked LS account(s) he would like to pull additional accounts and attributes from
6. If the user chooses not to use aggregation then IdP(a) returns a <samlp:Response> message containing a single SAML assertion, which holds both the principals authentication and attribute statements, to the LS. If however the principal wishes to use attribute aggregation then IdP(a) returns a new <Samlp:Response> message containing three SAML statements to the LS as described in section 3.1.
7. The LS attempts to identify the user account referenced in the referral. If the referral is valid at the LS and the LS is configured not to aggregate additional attributes, then the LS creates a new Attribute assertion containing additional referral information for each of the user's stored accounts and returns a <samlp:Response> to the user containing both the new attribute assertion and the

original authentication information. (See Section 3.1.2 for additional information and processing rules)

8. The SP then processes the authentication and attribute assertions. If the attributes returned are sufficient to authorise the principal then the process ends and the user is granted access to the resource.
9. If insufficient attributes are returned to authorise the principal then the SP processes the assertion generated by the LS and sends a new soap message to each of the entities identified in the returned EPRs. If an endpoint reference refers to a discovery service entity then the discovery service is queried with an IDWSF <disco:Query> message requesting additional user EPRs, if however the endpoint reference refers to a SAML2 Attribute Authority endpoint then a SAML2 <samlp:AttributeQuery> is generated using the initial random identifier as the request's subject. As the response has come from an LS entity any discovered EPRs will be for Discovery service entities and therefore each one will be queried with a <disco:Query> request for additional discovery service or attribute authority endpoints. As in this profile the SP is to perform aggregation the "aggregate" attribute of the request will be set to false.
10. The queried IdP then processes the discovery query request and maps the Subject of the Authentication statement issued by the authenticating IdP and contained in one of the request's <sec:Token> elements to the persistent identifier contained in the referral. If the queried IdP trusts that the initial IdP has authenticated the user sufficiently then a DiscoveryResponse message is created containing a single EPR pointing to the address of the SAML 2.0 attribute authority at which the mapped random identifier is now valid.
11. The SP processes the returned messages as follows, Any EPR's contained in the returned QueryResponse messages are processed and the EPR service type is checked. If the endpoint reference refers to a discovery service then the discovery service is queried as before, if however the endpoint reference refers to a SAML2 Attribute Authority endpoint then a SAML2 <samlp:AttributeRequest> is generated using the initial random identifier as the request's subject.
12. The IdP then processes the attribute query having mapped the random identifier used in the attribute request to the persistent identifier in step 10. A soap message is then created containing a samlp:Response> message holding the principal's attributes. This <samlp:Response> is then signed and returned by the IdP to the LS. The <samlp:Response> contains a single attribute assertion who's <saml:Subject> element matches that of the initial authentication assertion. The SP processes the <samlp:Response> message and decrypts the attributes in the returned assertion and uses the combined set of returned attributes for authorisation.

3.4 IdP Direct LS Aggregation

IdP First LS Aggregation.


User Agent
Service Provider
IdP A (Authn/AA)
Linking Service
IdP B (Disco)
IdP B (AA)
1. User Attempts to access a protected service at the SP
2. <samlp:AuthnRequest> message
3. The User authenticates him/herself to the IdP (Out of Scope)
4. <samlp:Response> containing SSO token, atts and EPR (referral 1)
5. If the attributes are sufficient then GRANT.
6 <disco:Query> message to the LS (referral 1).
7. <disco:Query> asking to map identifier and return IdP B's AA
8. <disco:QueryResponse> containing IdP B's AA EPR
9. <samlp:AttributeQuery> requesting user's attributes
10. <samlp:Response> containing User attributes encrypted to SP
11. <disco:QueryResponse> containing the EPR of the LS's AA endpoint
12. <samlp:AttributeQuery> requesting user's attributes
13. <samlp:Response> containing User attributes from IdP B encrypted to SP
14. GRANT/DENY

1. The user requests a service at the SP.
2. The SP sends a <samlp:AuthnRequest> protocol message via the users browser to IdP(a). (See Section 3.1.1 for additional information and processing rules)
3. Authentication is performed at IdP(a) in the usual way except for one small change in that during authentication the user is also asked whether or not he wishes to return additional attributes and if so he is also asked which linked LS account(s) he would like to pull additional accounts and attributes from
4. If the user chooses not to use aggregation then IdP(a) returns a <samlp:Response> containing a single SAML assertion which contains both the principals authentication and attribute statements to the SP. If however the principal wishes to use attribute aggregation then IdP(a) returns a new <Samlp:Response> message containing three SAML statements to the SP as described in section 4.2.
5. The SP then processes the authentication and attribute assertions. If the attributes returned are sufficient to authorise the principal then the process ends and the user is granted access to the resource.
6. If insufficient attributes are returned to authorise the principal then the SP processes the third statement containing additional identity account information and sends a new soap message containing an IDWSF Discovery Query Request to each of the LS entities identified in the returned EPR attributes.
7. The LS then processes the Discovery Query request, first using the encrypted persistent name identifier of the user contained in the security header to authenticate the request and to identify an internal account. Once this has been accomplished the authentication assertion is examined to determine whether the

service trusts the authenticating IdP and whether the authentication LoA is sufficient to release additional attributes to the user. The LS then tries to determine whether or not it should aggregate additional attributes itself. If it should then it constructs a new Discovery Query protocol message requesting the attribute authority endpoint of each IdP.

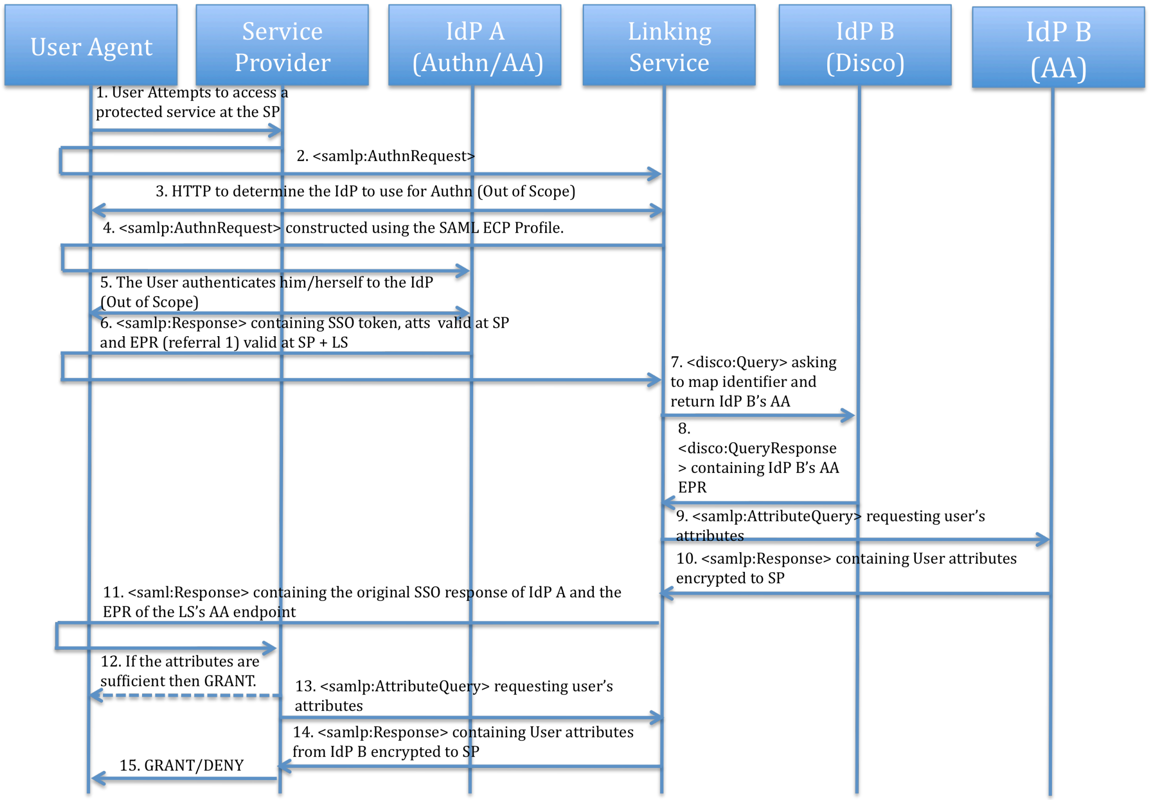
8. The queried IdP then processes the discovery query request and maps the Subject of the Authentication statement issued by the authenticating IdP and contained in one of the request's <sec:Token> elements to the persistent identifier contained in the referral as in step 7. If the queried IdP trusts that the initial IdP has authenticated the user sufficiently then a DiscoveryResponse message is created containing a single EPR containing the address of the SAML 2.0 attribute authority at which the mapped random identifier is now valid.
9. The LS processes the returned messages as follows, Any EPR's contained in the returned QueryResponse messages are processed and the EPR service type is checked. If the endpoint reference refers to a discovery service then the discovery serice is queried as before, if however the endpoint reference refers to a SAML2 Attribute Authority endpoint then a SAML2 <samlp:AttributeRequest> is generated using the initial random identifier as the request's subject.
10. The IdP then processes the attribute query having mapped the random identifier used in the attribute request to the persistent identifier in step 8. A soap message is then created containing a <samlp:Response> message holding the principal's attributes. This <samlp:Response> is then signed and returned by the IdP to the LS. The subject of this <samlp:Response> MUST be the random identifier generated in the original authentication response. The <samlp:Response> should contain a single attribute assertion who's <saml:Subject> element matches that of the initial authentication assertion. This attribute assertion MUST be encrypted to the SP specified in the AssertionConsumerServiceURL attribute of the attribute request and signed by the IdP.
11. When the LS has received a response from each queried IdP. It responds to the intial IDWSF Discovery Query message. This QueryResponse message contains a single EPR that points to the attribute authority endpoint of the LS. If no assertions have been returned to the LS then it may choose to return an empty response to prevent the SP from querying it for attributes.
12. The SP processes the <disco:QueryResponse> message and processes the returned messages as follows, Any EPR's contained in the returned QueryResponse messages are processed and the EPR service type is checked. If the endpoint reference refers to a discovery service then the discovery service is queried as before, if however the endpoint reference refers to a SAML2 Attribute Authority endpoint then a SAML2 <samlp:AttributeRequest> is generated using the initial random identifier as the request's subject
13. The LS then processes the attribute query having mapped the random identifier used in the attribute request to the persistent identifier in step 11. A soap message is then created containing a <samlp:Response> message holding each of the assertions returned to the LS by the queried IdPs. This <samlp:Response> is then signed and returned by the LS to the SP. The subject of this <samlp:Response> MUST be the random identifier generated in the original authentication response.

The <samlp:Response> may contain multiple attribute assertions who's <saml:Subject> elements all match that of the initial authentication assertion. This attribute assertion MUST be encrypted to the SP specified in the AssertionConsumerServiceURL attribute of the attribute request and signed by the LS.

14. The SP processes the <samlp:Response> message and decrypts the attributes in all the returned assertions and uses the combined set of returned attributes for authorisation.

3.3 IdP via LS, LS Aggregation

IdP via LS, LS Aggregation


User Agent
Service Provider
IdP A (Authn/AA)
Linking Service
IdP B (Disco)
IdP B (AA)
1. User Attempts to access a protected service at the SP
2. <samlp:AuthnRequest>
3. HTTP to determine the IdP to use for Authn (Out of Scope)
4. <samlp:AuthnRequest> constructed using the SAML ECP Profile.
5. The User authenticates him/herself to the IdP (Out of Scope)
6. <samlp:Response> containing SSO token, atts valid at SP and EPR (referral 1) valid at SP + LS
7. <disco:Query> asking to map identifier and return IdP B's AA
8. <disco:QueryResponse > containing IdP B's AA EPR
9. <samlp:AttributeQuery> requesting user's attributes
10. <samlp:Response> containing User attributes encrypted to SP
11. <saml:Response> containing the original SSO response of IdP A and the EPR of the LS's AA endpoint
12. If the attributes are sufficient then GRANT.
13. <samlp:AttributeQuery> requesting user's attributes
14. <samlp:Response> containing User attributes from IdP B encrypted to SP
15. GRANT/DENY

1. The user requests a service at the SP.
2. The SP sends a <samlp:AuthnRequest> protocol message via the users browser to the LS. (See Section 3.1.2 for additional information and processing rules)
3. The LS determines the IdP to use for authentication.
4. The LS sends a new <samlp:AuthnRequest> protocol message via the users browser to the LS. (See Section 3.1.2 for additional information and processing rules)
5. Authentication is performed at IdP(a) in the usual way except for one small change in that during authentication the user is also asked whether or not he wishes to use aggregation to return additional attributes and if so he is also asked

which linked LS account(s) he would like to pull additional accounts and attributes from

6. If the user chooses not to use aggregation then IdP(a) returns a <samlp:Response> containing a single SAML assertion which contains both the principals authentication and attribute statements to the LS. If however the principal wishes to use attribute aggregation then IdP(a) returns a new <Samlp:Response> message containing three SAML statements to the LS as described in section 3.1.
7. The LS attempts to identify the user account referenced in the referral. If the referral is valid at the LS and the LS is to perform LS aggregation then it constructs a new Discovery Query protocol message requesting the attribute authority endpoint of each IdP.
8. The queried IdP then processes the discovery query request and maps the Subject of the Authentication statement issued by the authenticating IdP and contained in one of the request's <sec:Token> elements to the persistent identifier contained in the referral. If the queried IdP trusts that the initial IdP has authenticated the user sufficiently then a DiscoveryResponse message is created containing a single EPR containing the address of the SAML 2.0 attribute authority at which the mapped random identifier is now valid.
9. The LS processes the returned messages as follows, Any EPR's contained in the returned QueryResponse messages are processed and the EPR service type is checked. If the endpoint reference refers to a discovery service then the discovery serice is queried as before, if however the endpoint reference refers to a SAML2 Attribute Authority endpoint then a SAML2 <samlp:AttributeRequest> is generated using the initial random identifier as the request's subject.
10. The IdP then processes the attribute query having mapped the random identifier used in the attribute request to the persistent identifier in step 8. A soap message is then created containing a <samlp:Response> message holding the principal's attributes. This <samlp:Response> is then signed and returned by the IdP to the LS. The subject of this <samlp:Response> MUST be the random identifier generated in the original authentication response. The <samlp:Response> should contain a single attribute assertion who's <saml:Subject> element matches that of the initial authentication assertion. This attribute assertion MUST be encrypted to the SP specified in the AssertionConsumerServiceURL attribute of the attribute request and signed by the IdP.
11. Once each IdP has responed to the LS's request it constructs a <samlp:Response> message to the SP containing a new SSO token constructed according to the SAML Enhanced Client Proxying rules which contains a new EPR attribute that points towards the LS's attribute endpoint and the original SSO assertion containing the original set of user attributes.
12. The SP then process the authentication and attribute assertions. If the attributes returned are sufficient to authorise the principal then the process ends and the user is granted access to the resource.
13. If however insufficient attributes are returned to authorise the principal then the SP processes the assertion generated by the LS, containing the EPR of the LS's attribute authority and sends a new soap message containing a <samlp:AttributeQuery> to each of the entities identified in the returned EPRs. If

an endpoint reference refers to a discovery service then the discovery service is queried as before, if however the endpoint reference refers to a SAML2 Attribute Authority endpoint then a SAML2 <samlp:AttributeQuery> is generated using the initial random identifier as the request's subject.

14. The LS then processes the attribute query having mapped the random identifier used in the attribute request to the persistent identifier in step 11. A soap message is then created containing a <samlp:Response> message holding each of the assertions returned to the LS when aggregating the principal's attributes. This <samlp:Response> is then signed and returned by the LS to the SP. The subject of each assertion contained in this <samlp:Response> MUST be the random identifier generated in the original authentication response. The <samlp:Response> should contain multiple assertions who's <saml:Subject> element matches that of the initial authentication assertion. This attribute assertion MUST be encrypted to the SP specified in the AssertionConsumerServiceURL attribute of the attribute request and each assertion signed by their issuing IdPs.
15. The SP processes the <samlp:Response> message and decrypts the attributes in each of the returned assertions and uses the combined set of returned attributes for authorisation.

4. Protocol Message Profiles

Each Protocol message sent using this profile MUST utilise the constrained SAML and IDWSF protocol messages described in this section. For more details of how these message are utilised within the Shintau architecture please refer to Section 3 which describes the potential protocol models and describes each of their respective message flows.

4.1 SAML 2.0 <samlp:AuthnRequest> to a linking Service or Identity Provider

The <samlp:AuthnRequest> (Section 3.4.1 of [6]) messages profiled below are used to request that an IdP authenticate a user and return authentication and attribute information to the SP. The SP may query a LS entity using this request, in which case the LS should proxy the request using the ECP Proxy rules defined in Section 4.2 of [SAML-PROFILES].

A <samlp:AuthnRequest> used to authenticate a user for Shintau aggregation MUST be created according to the rules set out below:

- The request's optional <saml:Subject> element SHOULD not be defined.
- The NameIDPolicy element of the request SHOULD be defined as follows:
  - The Format attribute MUST be set to "urn:oasis:names:tc:SAML:2.0:nameid-format:transient". Thereby specifying that the queried IdP MUST return an opaque and temporary value
  - The SPNameQualifier attribute MUST be set to the SP.
  - The AllowCreate attribute SHOULD be set to true

- The saml:Conditions element MAY contain NotBefore or NotOnOrAfter attributes if the SP wishes to limit the validity of the response.
- The RequestedAuthnContext attribute MAY be specified if the SP requires a specific authentication method to be used but its use is not required.
- The Scoping element of the request MUST be used to specify the RequesterID if the query is to be proxied by the LS or an IdP to another IdP to perform authentication.
- The ForceAuthn attribute MUST be true requiring that the principal be freshly authenticatated rather than allowing the use of pre-existing authentication data.
- The IsPassive attribute MUST not be defined
- The AssertionConsumerServiceURL attribute MUST be set to the issuer of the original authentication request, thereby specifying the ultimate consumer of the response. When proxying this attribute MUST be set to the SP.
- The AssertionConsumerServiceIndex attribute SHOULD not be specified
- The ProtocolBinding attribute is not specified in this document but MAY be included in the request.
- The AttributeConsumingServiceIndex attribute MUST be used to specify a set of attributes that the SP requires returned for authorisation as defined in the SP's metadata. If proxying a request the attributes requested MUST be identical to those provided in the original request.
- The ProviderName attribute MAY contain the human readable name of the requester.

e.g.

```
<samlp:AuthnRequest xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
ID="Request1" Version="2.0"  IssueInstant="2007-11-26T07:35:00Z " ForceAuthn=”true”
AssertionConsumerServiceURL=”sp.kent.ac.uk” AttributeConsumingServiceIndex=”1”
ProviderName=”Kent SP” >
    <saml:NameIDPolicy Format=” urn:oasis:names:tc:SAML:2.0:nameid-format:transient”
    SPNameQualifier=”sp.kent.ac.uk” AllowCreate=”true” />
    <saml:Conditions NotBefore=“"2007-11-26T07:35:00Z"  NotOnOrAfter="2007-11-
26T07:40:00Z"  />
</samlp:AuthnRequest>
```

4.2 IdP SAML 2.0 <samlp:Response> in response to a <samlp:AuthnRequest>

When an IdP receives the <samlp:AuthnRequest> it authenticates the user and conveys it response to the authentication request as a SAML 2.0 <samlp:Response> message (see section 3.3.3 of [6])

A <samlp:Response> message used to convey an authentication response for Shintau Aggregation MUST be created according to the rules set out below:

- If attribute aggregation is not to be used a standard <samlp:Response> containing an SSO response is constructed. This response message is not profiled in this document

- Otherwise if aggregation is enabled a <samlp:Response> message is constructed containing a single SSO assertion with three Statement elements.
    - The first of these statements is a new SSO authentication statement, which MAY contain a LOA attribute as the principals authentication context.
    - The second statement contains the user's attributes stored at the IdP that match the attributes requested in the <samlp:AuthnRequest>.
    - The third SAML statement should contain additional IDWSF Endpoint References (EPR) which specify how to access the linking service entities the principal wishes to use for aggregation and provide a security token for doing so (see 3.3). All of these EPR attributes are stored in a single attribute statement.
- The <saml:Subject> element of this assertion should be a new random identifier that corresponds to the <saml:NameIDPolicy> element of the request.
- The <samlp:Response> message MAY be encrypted using the SPs public key and SHOULD be signed using the IdPs private key.
- The SAML attribute statement containing user attributes within the response should specify the SP as their audience and MUST be encrypted to the SP and SHOULD be signed using the IdP's private key certificate.
- The SAML attribute statement containing EPR referrals should specify both the SP and the LS as their audience and SHOULD NOT be encrypted.

If the message is proxied via the LS then the LS should construct a response utilising the ECP proxying profile defined in (Section 4.2 of [SAML-PROFILES]) and may choose to include additional assertions containing additional EPR attributes.

4.3 Encoding a Referral as an IDWSF Endpoint Reference Attribute

In order to pass identity account information between different federated entities we utilise Liberty Alliance Endpoint Reference (EPR) attributes. (Section 4 of [7]). Before these messages are constructed we assume that the Issuer is already in possession of the following information:

- A persistent name identifier that is valid at the entity being referenced. It is also assumed that this identifier is valid only between the issuer and the referenced entity, allowing it to act as shared secret.
- The location of the service being referenced.
- The type of the service being referenced

EPRs and their corresponding <sec:Token> elements should be created according to the following rules:

- The attribute statement of the assertion containing additional identity information MUST only contain Liberty IDWSF endpoint reference attributes. Each of those EPR attributes MUST be created using the following definitions:
    - The wsu:Id attribute MUST be a unique reference that can be used to identify the EPR.
    - The reqRef attribute SHOULD not be present.

    - The notOnOrAfter attribute if present MUST represent the time after which the EPR should not be used.
    - The Abstract element MAY contain a textual natural description of the service instance defined in the EPR.
    - The ProviderID element MUST take the URI value of the provider of the service
    - The ServiceType element MUST take a value that represents a SAML V2.0 Authentication provider endpoint
    - the SecurityContext element MUST contain:
        - A single <SecurityMechID> element of value “urn:liberty:security:2005-02:TLS:SAML”, thereby specifying that the <sec:Token> element below is a SAML V2.0 assertion.
        - A single <sec:Token> element, containing a referral assertion (defined below) which can then be used to access the service described in the EPR
    - The framework element SHOULD take the value “2.0” thereby stating that the IDWSF 2.0 framework is being used.
    - The Actions element SHOULD NOT be defined
    - the options element SHOULD NOT be defined

Creation of a Referral:

Each assertion used to access user accounts (referrals) should be created using the following rules:

- The assertion’s Version attribute SHOULD be set to 2.0
- The ID attribute SHOULD be set to a unique identifier which can later be used to represent this assertion
- The IssueInstant attribute MUST be set to the time the Referral was issued
- The Issuer attribute MUST be set to the issuer of the Referral
- The ds:Signature element MUST contain the signature of the issuing party
- The assertions Subject MUST be an <EncryptedID> element. The decrypted value of this <EncryptedID> MUST contain a persistent account identifer valid between the issuing entity and the target of the referral.
- The Conditions element MUST contain:
    - An Audience restriction stating that the assertion is only valid when presented at the IdP/LS stated.
- The Advice element SHOULD contain:
    - An AssertionIDRef element that points to the authentication assertion used in the initial act of authentication to provide a link between the referral and the principals act of authentication.

4.4 IDWSF Discovery Query message Requesting additional EPR account information

IDSWF Discovery Query messages (Section 3.3.3 of [7]) are used by this profile to enable the requestor to determine additional endpoint references for use in the aggregation process. This query requests both discovery service endpoints and SAML 2.0 Attribute endpoints. In most case the query will be constructed using an existing EPR attribute and the original SSO token issued by the authenticating IdP.

A Discovery Query Request message constrained by this profile is constructed as follows:

- A non-standardised Boolean attribute “aggregate” is used to define whether the recipient should attempt to aggregate additional attributes or whether referrals should be returned to allow the requestor to perform the aggregation itself.
    - If “aggregate” is true then the recipient SHOULD attempt to aggregate additional attributes and return an endpoint reference from which the aggregated attributes can be returned to the requestor.
    - If “aggregate” is false or not present then the recipient SHOULD return any EPRs that can be used by the requestor to aggregate attributes.
        - If an IdP is queried using this message then it MUST return an attribute authority endpoint reference as it has no account information to return.

There is one <RequestedServiceType> element in which:

- There are two ServiceType elements, the first requests additional IDWSF discovery service EPRS and has a value of “urn:liberty:disco:2006-08” and the second requests additional SAML 2.0 attribute authority endpoints and has a value of “urn:oasis:names:tc:SAML:2.0:metadata:AttributeService”.
- There is a single <SecurityMechID> element that indicates that SAML assertions should be supplied as security tokens to access and discovered services. All other optional attributes should not be set.

The conveying SOAP message is then constructed according to section 5.10 of the Liberty ID-WSF SOAP Binding Specifications except for a single change in which both the original <sec:Token> element of the intial EPR and a new <sec:Token> element containing the authentication assertion are placed into the <Security> SOAP Header block for authentication purposes.

4.5 IDWSF Discovery QueryResponse message containing additional EPR account information

When a service is queried using an IDWSF Discovery Query message it MUST respond using a IDWSF Discovery QueryResponse message. If the queried entity is an LS it will either return a <disco:QueryResponse> message containing referrals to the discovery service endpoints of each IdPs where the user has additional account information or a

<disco:QueryResponse> containing an EPR referencing its own SAML 2.0 attribute authority endpoint. Discovery Service endpoints will be returned unless it has been indicated that the LS should perform aggregation itself. If the queried entity is an IdP then the service will always return an EPR referencing its own SAML 2.0 attribute authority endpoint.

The query response message is created as follows:

- The <lu:Status> element MUST have a value of "OK" unless there was a problem processing the request.
- If the service is an LS and is NOT performing aggregation there are no EndpointReference elements unless the LS wishes to share additional account identifiers with the SP in which case for each additional account a single EndpointReference element is created as discussed in section 3.4 above.
- If the Service is an LS and is performing aggregation or an IdP then there is a single EndpointReference element pointing to the queried service's SAML 2.0 Attribute Endpoint.
- The QueryResponse Message MAY be encrypted using the recipients public key certificate
- The QueryResponse Message MUST be signed using the issuers private key certificate.

4.6 SAML 2.0 <samlp:AttributeQuery> message to an IdP or LS for aggregated attributes

In order to query a SAML2 Attribute Authority endpoint entity for additional attributes a <samlp:AttributeQuery> message is utilised. This <samlp:AttributeQuery> is generated using the initial random identifier taken from the initial act of authentication as the request's subject.

The <samlp:AttributeQuery> is constructed as follows:

- The <saml:Subject> element of the query MUST contain the same random name identifier used in the initial authentication query issued by the authenticating IdP.
- The <saml:Attribute> element MAY contain a request for each of the attributes required to authorise the principal but may be omitted indicating that all attributes should be returned.
- Due to the potential for distributed aggregation that our conceptual models require, a need for a new attribute AssertionConsumerServiceURL has been found. This attribute i equivalent to the attribute of the same name found in the SAML V2.0 <AuthnRequest> in section 3.4.1 of [saml:CORE]. Its purpose here is to directly identify the ultimate consumer of the aggregated attributes i.e. the SP to allow the attributes to be encrypted using the SPs public key.

4.7 SAML 2.0 <samlp:Response> message in response to a <samlp:AttributeQuery>

When a service receives a <samlp:AttributeQuery> it must return a new <samlp:Response> message containing any attributes that could be determined in response to the request. If the queried party is an LS then the <samlp:Response> message MAY contain multiple attribute assertions each signed by their respective IdPs.

This response should be constructed in the following manner:

- The <samlp:Response> message MUST be signed using the issuers private key and MAY be encrypted using the public and returned by the IdP to the LS.
- The subject of this <samlp:Response> MUST be the random identifier generated in the original authentication response.
- If the <samlp:Response> MUST contain at least one attribute assertion each of which should be constrained in the following manner:
    - The <saml:Subject> element of each assertion MUST match that of the initial authentication assertion.
    - Each attribute assertion MUST be encrypted to the SP specified in the AssertionConsumerServiceURL attribute of the attribute query and signed by their respective IdPs.

5. Enhanced Linking Service Support for attribute aggregation

Each of the protocol models defined above has also been profiled for enhanced linking services clients to allow for greater user input in the aggregation process. This primarily consists of redirecting IDWSF discovery query requests via the user's browser/agent, instead of making requests directly to the Linking service. Both approaches have their advantages and their disadvantages back channel requests should be quicker and require less user involvement whereas front channel requests allow us to query the user for additional information during the aggregation process providing a more user centric experience. Most notably the LS may wish to ask the user directly (via a HTTP exchange) to choose which linked IdPs to aggregate from at Service Provision time rather than working from a static account release policy. When utilising the front channel bindings all existing protocol messages should be constructed in the same manner as before but transported via a HTTP redirect or similar.

6. Changes required to existing Infrastructure to implement protocols.

6.1 IdP direct SP Aggregation

In order to implement the IdP direct SP aggregation model discussed above I believe that the following changes would have to be made to the existing infrastructure:

The SP already posses the capacity to issue and process the required authentication and attribute queries/responses returned in the protocol. In order to support the IdP direct SP Aggregation model the SP would have to be altered in order for it to be able to process

EPR attributes stores in attribute assertions. The SP would also need to altered in order to allow it to issue SOAP messages containing IDWSF Discovery Query requests. The SP would also need to be able to process IDWSF Discovery QueryResponse messages. It must then be able to process these responses and be able to issue modeified <saml:AtrributeQuery> requests to the entities described in the EPR attributes it receives.

The LS would need to be able to accept the SP's IDWSF Discovery Query Request. The LS must then be able to map the encrypted persistent identity to a user account in a database containing linked principal accounts and accept the subsequent attribute request and recognise the aliased identifier. It must then be able to issue a new IDWSF QueryResponse message to the SP containing security tokens describing each of the additional accounts stored in its database.

The IdP would need to be changed in the following ways. A new listener capable of receiving and processing IDWSF Discovery Query messages would need to be implemented. In order to process these messages code would have to be added to allow the IdP to check the sec:Token contained in such a message against applicable attribute release policies and to allow the IdP to map the NameID element against a user in its database. Once the message has been processed the IDP would need code to issue an IDWSF QueryResponse message to the SP containing the AA of the service. The IdP would also have to be updated in order to store the alias contained in the Docivery Query message and to be able to check incoming AttributeRequest messages for the identifier and ultimately map the principals attributes to this new identifier. The authentication mechanism would also need to be changed to allow the principal to choose linking service entities to be included in the initial authentication response.

6.2 IdP direct LS Aggregation

In order to implement the IdP direct LS aggregation model discussed above I believe that the following changes would have to be made to the existing infrastructure:

The SP already posses the capacity to issue and process the required authentication and attribute queries/responses returned in the protocol. In order to support the IdP via LS SP Aggregation model the SP would have to be altered in order for it to be able to process EPR attributes stored in attribute assertions. The SP would also need to be altered to allow it to issue SOAP messages containing IDWSF Discovery Query requests. The SP would also need to be able to process IDWSF Discovery QueryResponse responses. It must then be able to process these responses and be able to issue additional IDWSF discovery query requests to the entities described in the EPR attributes it receives. It would also need to be altered to support the modified <samlp:AttributeQuery> message required by this profile.

The LS would need to be able to accept the SP's IDWSF Discovery Query Request. The LS must then be able to map the encrypted persistent identity to a user account in a database containing linked principal accounts and accept the subsequent attribute request and recognise the aliased identifier. It must then be able to issue IDWSF Discovery Query requests to each linked identity provider stored in its database and store the returned attribute assertions, before issuing a single soap message containing an IDWSF Query response message containing an EPR pointing to its own attribute authority endpoint. This AA endpoint must be able to support the modified <samlp:AttributeQuery> endpoint and map the subject of the request to an earlier mapped identifier before returning the full set of returned attribute assertions in a <samlp:Response> message.

The IdP would need to be changed in the following ways. A new listener capable of receiving and processing IDWSF Dicovery Query messages would need to be implemented. In order to process these messages code would have to be added to allow the IdP to check the sec:Token contained in such a message against applicable attribute release policies and to allow the IdP to map the NameID element against a user in its database. Once the message has been processed the IDP would need code to issue an IDWSF Discovery QueryResponse message to an appropriate consumer stating whether the request was successful and if so containing the EPR of the IdPs AA. The IdP would also have to be updated in order to store the alias contained in the IDWSF Dicovery Query and to be able to check incoming AttributeQuery messages for the identifier and ultimately map the principals attributes to this new identifier. The authentication mechanism would also need to be changed to allow the principal to choose linking service entities to be included in the initial authentication response.

6.3 IdP via LS SP Aggregation

In order to implement the IdP via LS SP aggregation model discussed above I believe that the following changes would have to be made to the existing infrastructure:

The SP already posses the capacity to issue and process the required authentication and attribute queries/responses returned in the protocol. In order to support the IdP direct SP Aggregation model the SP would have to be altered in order for it to be able to process EPR attributes stores in attribute assertions. The SP would also need to altered in order to allow it to issue SOAP messages containing IDWSF Discovery Query requests. The SP would also need to be able to process IDWSF Discovery QueryResponse messages. It must then be able to process these responses and be able to issue modified <saml:AtrributeQuery> requests to the entities described in the EPR attributes it receives.

The LS would need to be able to accept the SP's AuthnRequest message and support the SAML authentication proxying rules as set out in [saml:core]. It would also need to be able to issue a new SAML V2.0 AuthnRequest messages and be able to process the response accordingly. The LS must also be able to issue proxied authentication assertions to the SP. It would also need to support the use of IDWSF Discovery Query Request messages and have the capacity to issue IDWSF Discovery Query Responses containing tokens to access additional identity provider resources. The LS must be able to map the encrypted persistent identifiers contained in IDWSF Dicovery Query messages to user accounts in a database containing linked principal accounts and issue IDWSF Discovery Query response messages containing EPRs that can be used to access the linked identity provider accounts stored in its database

The IdP would need to be changed in the following ways. A new listener capable of receiving and processing IDWSF Discovery Query messages would need to be implemented. In order to process these messages code would have to be added to allow the IdP to check the sec:Token contained in such a message against applicable attribute release policies and to allow the IdP to map the NameID element against a user in its database. Once the message has been processed the IDP would need code to issue an IDWSF Discovery QueryResponse message to an appropriate consumer at the requestor stating whether the request was successful. The IdP would also have to be updated in order to store the alias contained in the IDWSF Discovery query and to be able to check incoming AttributeQuery messages for the identifier and ultimately map the principals attributes to this new identifier. The authentication mechanism would also need to be changed to allow the principal to choose linking service entities to be included in the initial authentication response.

6.4 IdP via LS LS Aggregation

In order to implement the IdP via LS LS aggregation model discussed above I believe that the following changes would have to be made to the existing infrastructure:

The SP already posses the capacity to issue and process the required authentication query/response returned in this protocol. The SP would need to be updated to process the EPR containing the LS's attribute authority endpoint. In order to query this endpoint the SP would need to be updated to support the modified AttributeQuery message required by this profile.

The LS would need to be able to accept the SP's AuthnRequest message and support the SAML authentication proxying rules as set out in [saml:core]. It would also need to be able to issue new SAML V2.0 AuthnRequest messages and be able to process the response accordingly. The LS must also be able to issue proxied authentication assertions to the SP. It would also need to support the use of IDWSF Discovery Query messages and have the capacity to issue IDWSF Discovery QueryResponse messages containing tokens to access additional identity provider resources. The LS must be able to map the encrypted persistent identifiers contained in the SOAP headers of the Discovery Query messages to user accounts in a database containing linked principal accounts. It must then be able to issue a soap message to each linked account rovider containing an IDWSF Discovery Query message requesting the Service's AA EPR and process the response. If the response contains EPRs to discovery endpoints then the LS should query those EPRs for additional EPRs. If however the response contains EPRs pointing to a SAML AA then the service should issue a <samlp:AttributeQuery> requesting attributes for the SP. It must then be able to store the returned attribute assertions until the service is queried for the attributes. Once each IdP has replied to the LS it should return an IDWSF Discovery QueryResponse message to the SP. The LS must also be able to accept a <samlp:AttributeQuery> message constructed according to the rules set out in this document and respond with a <samlp:Response> message containing the SAML assertions returned from the IdPs.

The IdP would need to be changed in the following ways. A new listener capable of receiving and processing IDWSF Discovery Query messages would need to be implemented. In order to process these messages code would have to be added to allow the IdP to check the sec:Token contained in such a message against applicable attribute release policies and to allow the IdP to map the NameID element against a user in its database. Once the message has been processed the IDP would need code to issue an IDWSF Discovery QueryResponse message to an appropriate consumer at the requestor stating whether the request was successful. The IdP would also have to be updated in order to store the alias contained in the IDWSF Discovery query and to be able to check incoming AttributeQuery messages for the identifier and ultimately map the principals attributes to this new identifier. The authentication mechanism would also need to be changed to allow the principal to choose linking service entities to be included in the initial authentication response.

7. Conceptual Model Support

As shown in the above protocol flows this protocol model can easily be modified to closely match each of the conceptual models discussed in the conceptual design document as it allows the user to pass an alias to an IdP inside the IDWSF Discovery Query message's SOAP header. The IdP can then use this to map a random identifier to a stored identifier which creates a session that can be accessed when an attribute query is recieved. It is slightly more complicated than the initial model as requires the use of two separate message types to create the desired result. This model can also be extended to convey additional referral information in the discovery query response generated by the IdPs allowing the LS or SP to discover additional linked IdP and SP entities if required.

8. Requirements Support

We believe that this protocol closely matches our initial set of requirements as it is likely to usable by a variety of applications due to its simplicity and lack of user interaction requiring only that the user chose an IdP and authenticate to it. It supports the privacy protection of user attributes by enabling principal identifiers and attributes to be encrypted to the SP preventing the LS entity from viewing user attributes. If the SP needs to track a user between sessions it can contact the LS and implement controls to track a user between sessions but can only discover the true identity of a user by contacting the principals initial IdP via the LS entity. A separate linking protocol ensures that users can only link attributes with the user's permission. This model utilizes the IDWSF SSO profile for secure soap transfer of SAML and Identity mapping messages allowing it to tunnel through firewalls. This protocol is entirely SAML and IDWSF compliant. The proxying of information can potentially be supported by this model. All entities have the ability to sign and encrypt assertions. This model has a low level of user interaction only requiring that the user to choose an IdP and then authenticates to that entity.

References

[1] Tom Barton, Jim Basney, Tim Freeman, Tom Scavo, Frank Siebenlist, Von Welch,

Rachana Ananthakrishnan, Bill Baker, Kate Keahey. “Identity Federation and Attribute-based Authorization through the Globus Toolkit, Shibboleth, GridShib, and MyProxy”. NIST PKI Workshop, April 2006
[2] Robinson, John-Paul. “MyVOCS: A Distributed Collaboration System”. Presentation available from http://www.stonesoup.org/Meetings/0609/vo.pres/robinson.pdf
[3] David Chadwick, George Inman, Nate Klingenstein. “Authorisation using Attributes from Multiple Authorities – A Study of Requirements”, EIfEL ePortfolio Conference, October 2007.
[4] David Chadwick, George Inman, Nate Klingenstein. “Conceptual Model for Attribute Aggregation”. Available from http://sec.cs.kent.ac.uk/shintau/Conceptual_Model.doc
[5] Scott Cantor. “Shibboleth Architecture, Protocols and Profiles, Working Draft 10 September 2005, see http://shibboleth.internet2.edu/shibboleth-documents.html
[6] OASIS, “Assertions and Protocols for the Security Assertion Markup Language (SAML) V2.0”, OASIS Standard, 15 March 2005
[7] Liberty Alliance, “Liberty ID-WSF Discovery Service Specification”, Liberty Alliance standard, v2.0
[8] Globus Toolkit v4
[9] OASIS, “Profiles for the OASIS Security Assertion Markup Language (SAML)V2.0” OASIS Standard, 15 March 2005.
[10] Liberty Alliance. “Liberty ID-WSF Security Mechanisms Core” Version 2.0
[11] CardSpace