Solution Proposal

We propose to combine the following SAML and Liberty Alliance profiles to address this use case:

- SAML 2.0 Web Browser SSO Profile
  - This is the basic Shibboleth solution and will bootstrap the second profile by decorating the assertion returned with additional content to enable delegated use.
- ID-WSF Enhanced Client or Proxy SSO Profile
  - This a profile for SAML-based authentication that is suited to non-browser software and can be related to the initial SSO event to create an auditable chain of activity.
- ID-WSF SAML Token Service Profile
  - This is a back-channel profile between a client and an Identity Povider to allow a client to explicitly request an assertion.
- SAML V2.0 Condition for Delegation Restriction
  - This is an extension for codifying the inclusion of delegation chains in SAML assertions in a mandatory-to-process fashion.

Properly composed, these specifications and their dependencies are sufficient to address the use case. SAML includes a basic ECP profile that is very similar to the Liberty version referenced above, but was designed for passing through browser-friendly login UI to a client, and not for server-side authentication to an IdP. The Liberty version relies on SOAP and a set of Liberty specifications and OASIS standards between the client (in this case the Portal/Portlet) and the IdP, which is a better framework for server-side authentication.

The full extent of the ID-WSF specifications, even limited to the set required to implement these profiles, are more extensive than what is required for our purposes. A separate topic discusses Suggested Profiling.

High-Level Solution

The solution recursively performs SAML SSO between the browser, Portal/Portlet, and a Web Service Provider. The second SSO operation with the WSP relies on the ECP variant rather than the standard browser version. In the middle of the two SSO steps, an optional request/response step can be inserted to act as a bridge between the two sequences if the Portal and Portlets are to be treated distinctly.

A high-level description of the token issuance sequence is something like this:

1. Browser given bearer assertion to login to Portal, includes an additional confirmation allowing the Portal to act as the user at the IdP.
2. Portal presents that assertion, and is given bearer assertion to hand to Portlet, includes an additional confirmation allowing the Portlet to act as the user at the IdP.
3. Portlet presents that assertion, and is given bearer assertion to hand to WSP. May include an additional confirmation allowing the WSP to act as the user at the IdP.

At each step, the IdP "enriches" the token for presentation back to itself (subject to whatever limits are imposed on the length of the chain). This enrichment takes the form of an additional subject confirmation, which can be either "bearer" or "holder-of-key", based on the desired level of security for the impersonation back to the IdP.

A bearer confirmation to the IdP is limited by time and client address, and locked to a specific endpoint at the IdP. A key confirmation would be limited by time and include a public key or certificate pulled from the relying party's metadata.

In all of the tokens, the subject and the statements refer to the user, and never the delegate systems. The delegates are identified in a mandatory-to-process SAML condition.

Solution Sequence

1. Browser SSO to Portal

The first set of steps consists of ordinary SAML Browser SSO (the "typical" Shibboleth use case), but notes some possible additions/optimizations that relate to subsequent processing.

The main difference is that the response to the SP (in this case the Portal) will include an additional assertion, or additional `<saml:SubjectConfirmation>` and `<saml:Audience>` elements in an existing assertion, that allows the Portal/Portlet to authenticate back to the IdP on the user's behalf. Essentially, the IdP issues a token that it can consume itself later. The duration of this capability can be tuned independently of other sessions, and can even be long lived for user-offline scenarios.

1.1. User Agent Issues HTTP Request to Portal

See step 1 of the Web Browser SSO Profile (section 4.1.3.1). This is the initial access to the portal (or possibly to a user interface element that initiates a login).

1.2. Portal Determines Identity Provider

See step 2 of the Web Browser SSO Profile (section 4.1.3.2). This may involve user interaction, cookies, and/or profiles involving IdP Discovery, but is not prescribed. Included here for completeness, but also to contrast with later steps in which the IdP can be deduced.

1.3. Portal Issues `<samlp:AuthnRequest>` to Identity Provider

See step 3 of the Web Browser SSO Profile (section 4.1.3.3). Various cues MAY be included in this request from the portal to optimize the rest of the flows. For example, one or more `<saml:Audience>` elements can be included to indicate that the Portal or Portlet(s) wish to obtain an assertion they can use as a delegate for the user.

Example

1.4. Identity Provider Identifies Principal

See step 4 of the Web Browser SSO Profile (section 4.1.3.4). This is the user authentication step, and likely includes SSO capability once the user has signed on.

1.5. Identity Provider Issues `<samlp:Response>` to Portal

See step 5 of the Web Browser SSO Profile (section 4.1.3.5).

Depending on the request content in step 1.3, or IdP configuration, the response MAY contain additional assertions, or the SSO assertion(s) targeted at the Portal MAY contain additional content enabling their use directly as delegated tokens or as authenticators to the IdP.

Example

1.6. Portal Renders Itself to User Agent

See step 6 of the Web Browser SSO Profile (section 4.1.3.6). This completes the process of authenticating the User Agent to the Portal, and ends the "typical" SAML flow.

2. Portlet Authentication and Access to Web Service Provider

The final set of steps is a recursive use of SAML SSO, as profiled by Liberty, to authenticate a Portlet to a Web Service Provider (WSP) that supports the SAML ECP SSO Profile.

2.1. Portlet Issues HTTP Request to Web Service Provider

See step 1 of the ID-WSF ECP SSO Profile (section 6.3.2). This is the initial access to the WSP without assuming any prior contact or security context.

Example

2.2. Web Service Provider issues `<samlp:AuthnRequest>` to Identity Provider via Portlet

See step 2 of the ID-WSF ECP SSO Profile (section 6.3.2). This is a PAOS request from the WSP to the Portlet that contains its request for SAML authentication. As noted in the profile, in some cases the Portlet could generate this request on its own, prior to issuing any requests to the WSP.

Example

2.3. Portlet Determines Identity Provider and Obtains Credential

Nominally corresponding to step 3 of the ID-WSF ECP SSO Profile (section 6.3.2), this step includes both the function of determining which IdP will need to be contacted and obtaining (in most cases) a SAML assertion that it can use to authenticate to the IdP in the following step on the user's behalf.

Determining the IdP is a function of examining the security context of the user (which will include that IdP), as well as verifying that the WSP does not preclude the use of that IdP via a SOAP header block in the request in step 2 above.

The Portlet must also obtain knowledge about where to contact that IdP and how it should secure its request in step 2.4 below. This information is represented in ID-WSF by a WS-Addressing EndpointReference, or EPR. An EPR is generally carried in a SAML attribute in the assertion issued by the IdP in step 1.5 above.

Obtaining the assertion to use when contacting the IdP is more complex, and is the primary distinction between a use case involving portlets and a more straightforward situation that does not involve a portal. With a non-portal use case, the initial response during SSO in phase 1 will generally contain a token that the application can use, so "obtaining the credential" is a simple matter of retrieving the token from the user's security context.

In a portal deployment, there are additional considerations, which we discuss in a separate topic, Token Acquisition by Portlets.

2.4. Portlet Forwards `<samlp:AuthnRequest>` to Identity Provider

See step 4 of the ID-WSF ECP SSO Profile (section 6.3.2). This is a SOAP request to the IdP, governed by additional ID-WSF profiles of SOAP and WS-Security that allow a SAML assertion to be used interoperably to authenticate the Portlet to the IdP on the user's behalf.

We do not formally constrain the possibilities for authenticating this exchange, but expect that the most likely security mechanisms used will be `urn:liberty:security:2006-08:ClientTLS:peerSAMLV2` and `urn:liberty:security:2006-08:TLS:SAMLV2`. In both cases, the Portlet uses a private key to satisfy the IdP of its right to use the authenticating assertion, but in the former case mutual TLS authentication is used, while in the latter case the SOAP message is digitally signed.

Example

2.5. Identity Provider Identifies Principal

See step 5 of the ID-WSF ECP SSO Profile (section 6.3.2). This is the "Portlet as User" authentication step, and processes the Portlet's request for a delegated authentication token to use at the WSP that issued the `<samlp:AuthnRequest>`. Policy may be applied here to control the issuance of such tokens, and XML Encryption can be used to hide all user information from the Portlet.

2.6. Identity Provider Issues `<samlp:Response>` to Web Service Provider via Portlet

See step 6 of the ID-WSF ECP SSO Profile (section 6.3.2). This is a SOAP response to the Portlet containing the SAML response targeted to the WSP containing the delegated authentication token with which the Portlet will be able to authenticate to the WSP as the user.

Example

Note that this delegated token may itself contain the capability to authenticate back to the IdP via a recursive act of delegation. This is orthogonal to the delegation that takes place **to** the WSP.

Example

2.7. Portlet Forwards `<samlp:Response>` to Web Service Provider

See step 7 of the ID-WSF ECP SSO Profile (section 6.3.2). This delivers the delegated authentication token to the WSP as a bearer assertion, analagous to the process by which a browser User Agent authenticates the user to a web site. A WSP can assess the act of delegation and respond appropriately, or ignore the distinction.

Example

2.8. Web Service Provider Responds to Portlet

See step 8 of the ID-WSF ECP SSO Profile (section 6.3.2). This step establishes a security context for the Portlet, probably via a cookie, and recovers the initial state of the request in step 1, if possible, in order to direct the Portlet to retry the request.