Protocol Support for New IdPs

Community Review in progress!

This document contains DRAFT material intended for discussion and comment by the InCommon participant community. Comments and questions should be sent to the InCommon participants mailing list (participants@incommon.org).

Recommended Protocol Support for New IdPs

Generally speaking, a good rule-of-thumb for new IdPs is to start simple and add more features and capabilities as the IdP matures and specific needs develop. Experience has shown that seldom used features are often deployed without adequate testing, leading to latent deployment bugs and even security holes.

The following deployment strategy forces all protocol traffic over the front channel, which is easier to troubleshoot, manage, and maintain.

Recommended Protocol Support for New IdPs

- **DO** support SAML2 Web Browser SSO on the front channel
- **DON'T** support back-channel SAML protocols

Later, if an SP partner requires the use of a back-channel SAML protocol, a new endpoint is easily added to metadata. However, since all new SPs registered in the Federation today are required to support SAML2 Web Browser SSO on the front channel, you may never need these extra SAML features.

An IdP should try to avoid SAML1 if possible, but in any case note the following:

- An IdP can support IdP-initiated SAML1 flows without advertising **any** SAML1 endpoints in metadata. (The same is true of SAML2 of course.)
- An IdP supports SP-initiated SAML1 flows by advertising a `SingleSignOnService` endpoint that supports the proprietary Shibboleth `AuthnRequest` binding/protocol.
- If your IdP must support SAML1, try to avoid SAML1 attribute query if at all possible. Consider pushing attributes via unencrypted SAML1 assertions on the front channel in lieu of attribute query.

All new IdPs MUST support SAML2 Web Browser SSO. Note the following specific recommendations:

- An IdP supports SP-initiated SAML2 flows by advertising a `SingleSignOnService` endpoint that supports the SAML2 `HTTP-Redirect` binding. Support for other bindings is optional and new deployments are encouraged to be conservative in this respect.
- An IdP should avoid advertising a SAML2 `AttributeService` endpoint in metadata. (An IdP deployment that routinely pushes attributes does not need to support SAML2 attribute query and doing so might cause spurious errors at the SP.)
- An IdP should push attributes via encrypted SAML2 assertions whenever possible. In practice, expecting all SP software to fully support XML Encryption is unreasonable, especially for non-Shibboleth SP software, so IdP operators should be prepared to make exceptions. That said, end-to-end encryption should remain a goal.

To illustrate the above recommendations in terms of metadata, here is a sample entity descriptor for an IdP that supports SAML2 Web Browser SSO on the front channel *only*:

```
<!-- The Example State University (example.edu) -->
<md:EntityDescriptor entityID="https://websso.example.edu/idp" xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata">
  <md:IDPSSODescriptor errorURL="https://login.example.edu/support.html" protocolSupportEnumeration="urn:oasis:
names:tc:SAML:2.0:protocol">
    <md:Extensions>
      <shibmd:Scope xmlns:shibmd="urn:mace:shibboleth:metadata:1.0" regexp="false">example.edu</shibmd:Scope>
      <mdui:UIInfo xmlns:mdui="urn:oasis:names:tc:SAML:metadata:ui">
        <mdui:DisplayName xml:lang="en">Example State University Secure Web Login</mdui:DisplayName>
        <mdui:InformationURL xml:lang="en">https://login.example.edu</mdui:InformationURL>
        <mdui:Logo height="128" width="128" xml:lang="en">https://login.example.edu/images/IdP_Logo.png</mdui:
Logo>
      </mdui:UIInfo>
    </md:Extensions>
    <md:KeyDescriptor use="signing">
      <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <ds:X509Data>
          <ds:X509Certificate>
MIIDyTCCArGgAwIBAgIJAKivSalalUbnMA0GCSqGSIb...
          </ds:X509Certificate>
        </ds:X509Data>
      </ds:KeyInfo>
    </md:KeyDescriptor>
    <md:SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect" Location="
https://login.example.edu/idp/saml2/Redirect/SSO"/>
    <md:SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="https://login.
example.edu/idp/saml2/POST/SSO"/>
  </md:IDPSSODescriptor>
  <md:Organization>
    <md:OrganizationName xml:lang="en">The Example State University</md:OrganizationName>
    <md:OrganizationDisplayName xml:lang="en">Example State University</md:OrganizationDisplayName>
    <md:OrganizationURL xml:lang="en">http://www.example.edu</md:OrganizationURL>
  </md:Organization>
  <md:ContactPerson contactType="technical">
    <md:GivenName>Technical Services</md:GivenName>
    <md:EmailAddress>tech-services@example.edu</md:EmailAddress>
  </md:ContactPerson>
  <md:ContactPerson contactType="administrative">
    <md:GivenName>Administrative Services</md:GivenName>
    <md:EmailAddress>admin-services@example.edu</md:EmailAddress>
  </md:ContactPerson>
</md:EntityDescriptor>
```

Note the following protocol-related features of this entity descriptor:

- There is just one role descriptor, given by the `<md:IDPSSODescriptor>` element.
- The `protocolSupportEnumeration` XML attribute indicates SAML2 only.
- There are two `SingleSignOnService` endpoints that support the `HTTP-Redirect` and `HTTP-POST` bindings (both of which are front-channel bindings).

If you compare this metadata to the majority of IdP entity descriptors in InCommon metadata, you'll notice the following significant differences:

- Most IdP entity descriptors include **two** role descriptors, indicated by the `<md:IDPSSODescriptor>` and `<md:AttributeAuthorityDescriptor>` elements.
- Each role descriptor contains its own key descriptor, and moreover, the two key descriptors are identical.
- Most IdPs publish one or more back-channel, SOAP-based endpoints, including:
  - one or more `ArtifactResolutionService` endpoints
  - one or more `AttributeService` endpoints
- Many IdPs support SAML1, and moreover, publish a SAML1 `AttributeService` endpoint

Clearly the metadata for an IdP that *only* supports SAML2 on the front channel is much simpler. That simplicity translates into an IdP that is easier to troubleshoot, manage, and maintain.