NativeSPADFS

The SP includes so-called "RP" support for the WS-Federation protocol as profiled by Microsoft in their ADFSv1 product. The SP can act as a WS-Federation Passive Profile relying party in the same fashion that it supports SAML. All SP features not specific to the SAML protocol are supported equally for WS-Federation IdPs.

This page is not relevant to use of ADFSv2, which supports a subset of SAML 2.0. The CommercialInterop page has information on SAML interoperation with Microsoft's implementation (what little has been provided, anyway).

Metadata

Support for WS-Federation is currently provisioned and secured using the same metadata sources used for SAML. A profile of SAML metadata for use by WS-Federation peers was developed for the Shibboleth 1.3 release and remains supported in Shibboleth 2.0.

- SAML metadata profile for WS-Federation

So the first step in enabling this support is to obtain or create metadata for the IdP following the profile. Without it, nothing will happen when you try to use the SP's features, or you'll get a metadata-related error.

Activating the ADFS Extension

The WS-Federation code for the SP is supplied in an extension that is included with the SP source code, and is built by default and included with binary packages. The extension is contained in two libraries, one for `shibd` and one for the web server filter/module. These are named `adfs.so` and `adfs-lite.so` respectively.

To load the extension, each library must be added to the `<Extensions>` element in the `<OutOfProcess>` and `<InProcess>` elements like so (the example isn't complete, only the relevant parts are shown):

```
<OutOfProcess>
    <Extensions>
        <Library path="adfs.so" fatal="true"/>
    </Extensions>
</OutOfProcess>
<InProcess>
    <Extensions>
        <Library path="adfs-lite.so" fatal="true"/>
    </Extensions>
</InProcess>
```


Extensions are **not** loaded dynamically. You will need to restart the relevant services after adding the above configuration

Enabling the WS-Federation Protocol (SP V2.4 and Above)

To enable the WS-Fed support on current SP versions, simply add the `ADFS` protocol token to the content of the `<SSO>` element (and if desired, the `<Logout>` element).

Enabling the WS-Federation Protocol (SP Versions < V2.4)

On older versions, enabling the plugin requires some simple modifications to the handlers defined inside the `<Sessions>` element:

- Add an `<md:AssertionConsumerService>` with `Binding="http://schemas.xmlsoap.org/ws/2003/07/secext"` to the list of endpoints. The `index` attribute is generally not important but should be unique.

```
<md:AssertionConsumerService Location="/ADFS" index="10" Binding="http://schemas.xmlsoap.org/ws/2003/07/secext"
/>
```

- Add a `<SessionInitiator>` with `type="ADFS"` to one or more of your initiator chains.

```
<!-- If outside of a chain, add Location="/Login" -->
<SessionInitiator type="ADFS" acsIndex="10"/>
```

It should be placed alongside or in place of the `"Shib1"` and `"SAML2"` plugins, in order of protocol preference. The `acsIndex` property should match your ACS handler's `index` from the previous step.

- If you want to support SP-initiated logout using the WS-Federation signout protocol, then add a `<LogoutInitiator>` with `type="ADFS"` to one or more of your logout chains, ahead of the element with `type="Local"`.

```
<!-- If outside of a chain, add Location="/Logout" -->
<LogoutInitiator type="ADFS"/>
```

Attribute Handling

As with most commercial SAML code, ADFS is a bit wonky in its support for SAML attributes. While Shibboleth makes no hardwired assumptions about attribute naming, most commercial code does. In the case of ADFS, a handful of built-in claims are included and any custom claims are generated with a proprietary AttributeNamespace value of `"http://schemas.xmlsoap.org/claims"`. On the SP side, interoperation therefore requires that custom entries be added to the attribute extraction configuration (typically `attribute-map.xml`).

To support ADFS claims passed as SAML attributes, you'll need to include the XML attribute/value of `nameFormat="http://schemas.xmlsoap.org/claims"` inside each `<Attribute>` element that specifies an ADFS claim.

```
    <!-- WS-Fed attributes -->
    <Attribute nameFormat="http://schemas.xmlsoap.org/claims" name="CommonName" id="cn"/>
    <Attribute nameFormat="http://schemas.xmlsoap.org/claims" name="EmailAddress" id="email"/>
    <Attribute nameFormat="http://schemas.xmlsoap.org/claims" name="UPN" id="userPrincipalName"/>
    <Attribute nameFormat="http://schemas.xmlsoap.org/claims" name="Group" id="group"/>
```

These claims can be added alongside the existing SAML definitions and map to the same internal `id` so that applications are unaware of the distinction.

Name Identifiers

To access information passed within the subject of the assertion, the special `nameFormat` is unnecessary. The `Format` of the `<saml:NameIdentifier>` is placed into the mapping `name`, per usual.