NameMapping

This page didn't survive the conversion process and is no longer very usable.

Zero or more `NameMapping` elements (in `idp.xml`) call out the name mappings recognized by a Shibboleth deployment. The `NameMapping` element supports the following attributes:

Error rendering macro 'html'

Notify your Confluence administrator that "Bob Swift Atlassian Add-ons - HTML" requires a valid license. Reason: EXPIRED

Note: One and only one of the `type` or `class` attributes is required.

A brief description of each attribute follows:

- `id`: a unique ID for this `NameMapping` element
- `format`: a NameIdentifierFormat associated with this `NameMapping` element
- `regex`: a regular expression used to extract the principal name from the DN in the `getPrincipal` method of class `X509SubjectNameNameIdentifierMapping`
- `qualifier`: a URI, which is matched against the value of the `NameQualifier` attribute (of the `<saml:NameIdentifier>` element) in the `getPrincipal` method of class `X509SubjectNameNameIdentifierMapping`
- `internalNameContext`: a string template containing one or more `%PRINCIPAL%` placeholders used to construct a `SAMLNameIdentifier` object in method `getNameIdentifierName` of class `X509SubjectNameNameIdentifierMapping`
- `handleTTL`: the time-to-live (TTL) of the handle in seconds
- `type`: an alias pre-registered with the `NameMapper` class (see NameIdentifierMapping for possible values)
- `class`: the fully qualified class name of an implementation of NameIdentifierMapping

A `NameMapping` element of type `CryptoHandleGenerator` (equivalent to class `CryptoShibHandle`) contains a number of child elements:

Error rendering macro 'html'

Notify your Confluence administrator that "Bob Swift Atlassian Add-ons - HTML" requires a valid license. Reason: EXPIRED

See the *Shibboleth Identity Provider Deployment Guide* for more detail regarding `CryptoShibHandle`. See http://java.sun.com/j2se/1.5.0/docs/guide/security/CryptoSpec.html for general information about cryptographic implementations, conventions and syntax.

Some examples of `NameMapping` elements are given below:

```
<!-- SharedMemoryShibHandle configuration (default) -->
<NameMapping
  xmlns="urn:mace:shibboleth:namemapper:1.0"
  id="..."
  format="urn:mace:shibboleth:1.0:nameIdentifier"
  handleTTL="1800"
  type="SharedMemoryShibHandle"/>

<!-- CryptoShibHandle configuration -->
<NameMapping
  xmlns="urn:mace:shibboleth:namemapper:1.0"
  id="..."
  format="urn:mace:shibboleth:1.0:nameIdentifier"
  handleTTL="1800"
  type="CryptoHandleGenerator">
  <KeyStorePath>...</KeyStorePath>
  <KeyStorePassword>...</KeyStorePassword>
  <KeyStoreKeyAlias>...</KeyStoreKeyAlias>
  <KeyStoreKeyPassword>...</KeyStoreKeyPassword>
  <KeyStoreType>JCEKS</KeyStoreType>  <!-- default -->
  <Cipher>DESede/CBC/PKCS5Padding</Cipher>  <!-- default -->
  <MAC>HmacSHA1</MAC>  <!-- default -->
</NameMapping>

<!-- PrincipalNameIdentifier configuration (test) -->
<NameMapping
  xmlns="urn:mace:shibboleth:namemapper:1.0"
  id="..."
  format="urn-x:test:NameIdFormat1"
  type="Principal"/>

<!-- X509SubjectNameNameIdentifierMapping configuration (e-auth) -->
<NameMapping
  xmlns="urn:mace:shibboleth:namemapper:1.0"
  id="..."
  format="urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName"
  regex=".*uid=([^,/]+).*"
  qualifier="https://idp.org/shibboleth"
  internalNameContext="uid=%PRINCIPAL%/e-auth"
  class="edu.internet2.middleware.shibboleth.common.provider.X509SubjectNameNameIdentifierMapping"/>
```

Only one `NameMapping` element per format is allowed. If you wanted to associate a single NameIdentifierFormat with multiple mappings, a custom `MappingManager` must be written.

```
<!-- hypothetical configuration (e.g.) -->
<NameMapping
  xmlns="urn:mace:shibboleth:namemapper:1.0"
  id="..."
  format="urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName"
  class="edu.uiuc.ncsa.shibboleth.plugins.MappingManager">
  <NameMapping
         id="..."
         format="urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName"
         regex=".*uid=([^,/]+).*"
         qualifier="https://idp.org/shibboleth"
         internalNameContext="uid=%PRINCIPAL%/e-auth"
         class="edu.internet2.middleware.shibboleth.common.provider.X509SubjectNameNameIdentifierMapping"/>
  <NameMapping
         id="..."
         format="urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName"
         class="edu.uiuc.ncsa.shibboleth.plugins.X509SubjectNameNameIdentifierMapping"/>
</NameMapping>
```

Presumably, the `MappingManager` invokes each of the nested mappings (in order) until the mapping succeeds.

For example, suppose an attribute query is sent to the AA with the following `NameIdentifier` element:

```
<saml:NameIdentifier
  Format="urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName"
  NameQualifier="https://idp.org/shibboleth">
  <!-- insert X.509 Subject DN here -->
</saml:NameIdentifier>
```

The AA consults `origin.xml` and finds a `NameMapping` element such as the last one above. Since the value of the `Format` attribute of the `NameIdentifier` element matches the value of the `format` attribute of the containing `NameMapping` element, the AA invokes the `MappingManager` as given by the `class` attribute. The `MappingManager` then applies each of the nested mappings in turn.

-- Main.TomScavo - 13 Apr 2005