Meta-Attributes

Meta-Attributes

A *meta-attribute* is an abstract "above-the-wire" user attribute. Meta-attributes are used to unambiguously specify attribute requirements in deployment profiles, in attribute queries, and in SAML metadata. The meta-attribute concept is similar to the notion of "scope" in OpenID Connect.

Contents

Meta-attributes provide an unambiguous language for formulating attribute requirements in service (or client) metadata or in attribute queries. Other uses of meta-attributes include:

- To standardize attributes (or claims) across multiple protocols (such as SAML Web Browser SSO and OpenID Connect)
- To reconcile apparent conflicts among entity categories with competing attribute requirements
- To provide flexibility to IdP operators when configuring attribute release policy rules
- To make it easy to design usable consent interfaces

For discussion purposes, a Meta-Attribute Registry is defined in the appendix below. The registry defines groups of attributes—including both eduPerson attributes and OpenID Connect claims—as higher-level meta-attributes. Using meta-attributes, references to user attributes can be made in schema-independent fashion.

Requested Attributes in Metadata

Each meta-attribute in the Meta-Attribute Registry includes an example that shows how the meta-attribute is used in (SAML) metadata. A more comprehensive example follows, including sample Shibboleth IdP V3 configurations based on requested attributes in SP metadata.

Requesting Wire Attributes in Metadata

Suppose an SP has the following requested attributes in metadata:

```
<md:RequestedAttribute FriendlyName="eduPersonPrincipalName"
   Name="urn:oid:1.3.6.1.4.1.5923.1.1.1.6"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"/>

<md:RequestedAttribute FriendlyName="displayName"
   Name="urn:oid:2.16.840.1.113730.3.1.241"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"/>

<md:RequestedAttribute FriendlyName="mail"
   Name="urn:oid:0.9.2342.19200300.100.1.3"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"/>
```

Then a Shibboleth IdP with the following configuration will release the indicated wire attributes to the above SP:

```
<afp:AttributeFilterPolicy id="releaseEssentialAttributesToAnySP">

  <afp:PolicyRequirementRule xsi:type="basic:ANY"/>

  <!-- assuming ePPN is non-reassigned -->
  <afp:AttributeRule attributeID="eduPersonPrincipalName">
    <afp:PermitValueRule xsi:type="AttributeInMetadata"/>
  </afp:AttributeRule>

  <afp:AttributeRule attributeID="displayName">
    <afp:PermitValueRule xsi:type="AttributeInMetadata"/>
  </afp:AttributeRule>

  <afp:AttributeRule attributeID="email">
    <afp:PermitValueRule xsi:type="AttributeInMetadata"/>
  </afp:AttributeRule>

</afp:AttributeFilterPolicy>
```

An IdP so configured will not release attributes to an SP unless the indicated requested attributes are in SP metadata.

Unfortunately, requesting specific wire attributes in this way doesn't work very well in practice. Take `eduPersonPrincipalName`, for instance. The SP might be perfectly happy to receive `eduPersonUniqueId` in lieu of `eduPersonPrincipalName` but unfortunately there is no way to express this complex requirement in metadata. This is where meta-attributes come in handy.

Requesting Meta-Attributes in Metadata

Now suppose an SP has the following requested attributes in metadata:

```
<md:RequestedAttribute FriendlyName="metaPublicUserID"
   Name="http://id.example.org/attribute/metaPublicUserID"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"/>

<md:RequestedAttribute FriendlyName="metaPersonName"
   Name="http://id.example.org/attribute/metaPersonName"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"/>

<md:RequestedAttribute FriendlyName="metaEmailAddress"
   Name="http://id.example.org/attribute/metaEmailAddress"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"/>
```

Then two Shibboleth IdPs each with the following configurations will release the indicated wire attributes to the above SP:

```
<afp:AttributeFilterPolicy id="
mapAndReleaseEssentialAttributesToAnySP">

  <afp:PolicyRequirementRule xsi:type="basic:ANY"/>

  <!-- assuming ePPN is non-reassigned -->
  <afp:AttributeRule attributeID="
eduPersonPrincipalName">
    <afp:PermitValueRule xsi:type="
AttributeInMetadata"
        attributeName="http://id.example.org
/attribute/metaPublicUserID"
        attributeNameFormat="urn:oasis:names:tc:
SAML:2.0:attrname-format:uri"/>
  </afp:AttributeRule>

  <afp:AttributeRule attributeID="displayName">
    <afp:PermitValueRule xsi:type="
AttributeInMetadata"
        attributeName="http://id.example.org
/attribute/metaPersonName"
        attributeNameFormat="urn:oasis:names:tc:
SAML:2.0:attrname-format:uri"/>
  </afp:AttributeRule>

  <afp:AttributeRule attributeID="email">
    <afp:PermitValueRule xsi:type="
AttributeInMetadata"
        attributeName="http://id.example.org
/attribute/metaEmailAddress"
        attributeNameFormat="urn:oasis:names:tc:
SAML:2.0:attrname-format:uri"/>
  </afp:AttributeRule>

</afp:AttributeFilterPolicy>
```

```
<afp:AttributeFilterPolicy id="
mapAndReleaseEssentialAttributesToAnySP">

  <afp:PolicyRequirementRule xsi:type="basic:ANY"/>

  <!-- where ePUId is non-reassigned by definition
-->
  <afp:AttributeRule attributeID="
eduPersonUniqueId">
    <afp:PermitValueRule xsi:type="
AttributeInMetadata"
        attributeName="http://id.example.org
/attribute/metaPublicUserID"
        attributeNameFormat="urn:oasis:names:tc:
SAML:2.0:attrname-format:uri"/>
  </afp:AttributeRule>

  <afp:AttributeRule attributeID="givenName">
    <afp:PermitValueRule xsi:type="
AttributeInMetadata"
        attributeName="http://id.example.org
/attribute/metaPersonName"
        attributeNameFormat="urn:oasis:names:tc:
SAML:2.0:attrname-format:uri"/>
  </afp:AttributeRule>

  <afp:AttributeRule attributeID="surname">
    <afp:PermitValueRule xsi:type="
AttributeInMetadata"
        attributeName="http://id.example.org
/attribute/metaPersonName"
        attributeNameFormat="urn:oasis:names:tc:
SAML:2.0:attrname-format:uri"/>
  </afp:AttributeRule>

  <afp:AttributeRule attributeID="email">
    <afp:PermitValueRule xsi:type="
AttributeInMetadata"
        attributeName="http://id.example.org
/attribute/metaEmailAddress"
        attributeNameFormat="urn:oasis:names:tc:
SAML:2.0:attrname-format:uri"/>
  </afp:AttributeRule>

</afp:AttributeFilterPolicy>
```

Note that both IdPs have an attribute release policy that relies on the same set of requested attributes, but the requested attributes are mapped to different wire attributes in each case.

Building a Better Entity Category

Let's see how meta-attributes can help us build an entity category that optimizes attribute release.

The primary purpose of a service category (i.e., a category of service providers) is to facilitate attribute release. Clearly IdPs won't release attributes to SPs just because there are requested attributes in metadata, but it may help. Consider the following hypothetical example of an entity category.

Let's begin by defining an attribute bundle consisting of just three meta-attributes:

1. *Public User Identifier*
2. *Person Name*
3. *Email Address*

Note that the underlying user attributes in the bundle are a subset of what is commonly called *directory information*.

Suppose the following entity attribute signifies membership in a hypothetical entity category we'll call the *Ready to Collaborate Category*:

```
<!-- the ready-to-collaborate entity attribute for SPs -->
<saml:Attribute xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
    NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
    Name="http://macedir.org/entity-category">
  <saml:AttributeValue>
    http://id.example.org/category/ready-to-collaborate
  </saml:AttributeValue>
</saml:Attribute>
```

An SP is a member of the Ready To Collaborate Category if it exhibits the `ready-to-collaborate` entity attribute in its metadata. An IdP that supports the Ready To Collaborate Category recognizes that entity attribute as follows:

```
<afp:AttributeFilterPolicy id="mapAndReleaseAttributesToAnyReadyToCollaborateSP">

  <afp:PolicyRequirementRule xsi:type="EntityAttributeExactMatch"
      attributeName="http://macedir.org/entity-category"
      attributeValue="http://id.example.org/category/ready-to-collaborate"/>

  <!-- assuming ePPN is non-reassigned -->
  <afp:AttributeRule attributeID="eduPersonPrincipalName">
    <afp:PermitValueRule xsi:type="AttributeInMetadata" onlyIfRequired="true"
        attributeName="http://id.example.org/attribute/metaPublicUserID"
        attributeNameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"/>
  </afp:AttributeRule>

  <afp:AttributeRule attributeID="displayName">
    <afp:PermitValueRule xsi:type="AttributeInMetadata" onlyIfRequired="true"
        attributeName="http://id.example.org/attribute/metaPersonName"
        attributeNameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"/>
  </afp:AttributeRule>

  <afp:AttributeRule attributeID="email">
    <afp:PermitValueRule xsi:type="AttributeInMetadata" onlyIfRequired="true"
        attributeName="http://id.example.org/attribute/metaEmailAddress"
        attributeNameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"/>
  </afp:AttributeRule>

</afp:AttributeFilterPolicy>
```

The IdP releases attributes to an SP when two conditions are met:

1. The SP is tagged with the `ready-to-collaborate` entity attribute (presumably put there by a federation operator)
2. The SP has one or more of the designated meta-attributes in its metadata

As shown earlier, different IdPs can release different wire attributes as long as the IdP conforms to the requirements of the category (which depends on meta-attributes in the registry).

The above configuration is optimal in the following sense:

- An attribute is released only if the corresponding `<md:RequestedAttribute>` element is decorated with an `isRequired="true"` XML attribute.
- If an SP lists all three meta-attributes in metadata, an IdP that supports the category will release all three.
- If an SP lists less than all three meta-attributes, then that's exactly what it gets.
- If an SP lists no meta-attributes in metadata, it should expect to receive **no** attributes from a supporting IdP.
- An SP may list more than the three meta-attributes in metadata, but a supporting IdP is not required to release them.

So the use of meta-attributes optimizes the attribute release process.

Appendix

Meta-Attribute Registry

A simple registry of meta-attributes illustrates the concept.

User Identifier

FriendlyName: `metaUserID`
Name: `http://id.example.org/attribute/metaUserID`

A `metaUserID` is a persistent, non-reassigned identifier.

An Identity Provider (or Attribute Authority) is said to *release a `metaUserID`* when it releases one of the following attributes on the wire:

1. `eduPersonTargetedID`
2. `eduPersonUniqueId`
3. `eduPersonPrincipalName` (if non-reassigned)
4. OpenID Connect `sub` claim

A Service Provider is said to *request a `metaUserID`* when it does so directly, as shown in the following example.

Example

Here is an example of an abstract `metaUserID` requested in Service Provider metadata:

```
<md:RequestedAttribute FriendlyName="metaUserID"
   Name="http://id.example.org/attribute/metaUserID"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"/>
```

Public User Identifier

FriendlyName: `metaPublicUserID`
Name: `http://id.example.org/attribute/metaPublicUserID`

A `metaPublicUserID` is a persistent, non-reassigned, non-targeted identifier.

An Identity Provider (or Attribute Authority) is said to *release a `metaPublicUserID`* when it releases one of the following attributes on the wire:

1. `eduPersonUniqueId`
2. `eduPersonPrincipalName` (if non-reassigned)
3. OpenID Connect public `sub` claim

A Service Provider is said to *request a `metaPublicUserID`* when it does so directly, as shown in the following example.

Example

Here is an example of an abstract `metaPublicUserID` requested in Service Provider metadata:

```
<md:RequestedAttribute FriendlyName="metaPublicUserID"
   Name="http://id.example.org/attribute/metaPublicUserID"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"/>
```

Person Name

FriendlyName: `metaPersonName`
Name: `http://id.example.org/attribute/metaPersonName`

A `metaPersonName` is a human-readable name for the person (or subject) involved in a federated transaction.

An Identity Provider (or Attribute Authority) is said to *release a `metaPersonName`* when it releases at least one of the following attributes (or attribute combinations) on the wire:

1. eduPerson `displayName` attribute
2. OpenID Connect `name` claim
3. Two eduPerson attributes: `givenName` + `sn` (surname)
4. Two OpenID Connect claims: `given_name` + `family_name`

A Service Provider is said to *request a `metaPersonName`* when it does so directly, as shown in the following example.

Example

Here is an example of an abstract `metaPersonName` requested in Service Provider metadata:

```
<md:RequestedAttribute FriendlyName="metaPersonName"
   Name="http://id.example.org/attribute/metaPersonName"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"/>
```

Email Address

FriendlyName: `metaEmailAddress`
Name: `http://id.example.org/attribute/metaEmailAddress`

A `metaEmailAddress` is an electronic mail address for the person (or subject) involved in a federated transaction.

An Identity Provider (or Attribute Authority) is said to *release a `metaEmailAddress`* when it releases one of the following attributes on the wire:

1. eduPerson `mail` attribute
2. OpenID Connect `email` claim

A Service Provider is said to *request a `metaEmailAddress`* when it does so directly, as shown in the following example.

Example

Here is an example of an abstract `metaEmailAddress` requested in Service Provider metadata:

```
<md:RequestedAttribute FriendlyName="metaEmailAddress"
   Name="http://id.example.org/attribute/metaEmailAddress"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"/>
```