NameIDFormatFilter

This feature requires V3.3 or later.

A filter of type `NameIDFormat` adds SAML `<md:NameIDFormat>` elements to metadata in order to drive software behavior (primarily Name Identifier format selection).

Sequences of string-valued `<Format>` elements are supplied as filter content. When a child element such as `<Entity>` or `<ConditionRef>` or `<ConditionScript>` evaluates to true, the formats are applied to all the recognized format-supporting roles of the corresponding entities. The filter does not have the capability to limit the roles to which formats will be attached.

Filter order is important!

This filter changes the content of the metadata and so a filter of type `NameIDFormat` should appear after any SignatureValidationFilter in the overall `MetadataProvider`.

Position the NameIDFormat filter for efficiency

Deliberately position a `NameIDFormat` filter in the overall sequence of filters for optimal efficiency. In particular, a filter of type `NameIDFormat` should appear after the EntityRoleWhiteListFilter since the latter effectively removes entities from the input.

Schema

The `<MetadataFilter>` element and the type `NameIDFormat` are defined by the `urn:mace:shibboleth:2.0:metadata` schema, which can be located at http://shibboleth.net/schema/idp/shibboleth-metadata.xsd

Reference

Attributes

Name	Type	Default	Description
`removeExistingFormats` 3.4	Boolean	false	Whether to remove any existing formats from a role if any are added by the filter (unmodified roles will be untouched regardless of this setting)

Child Elements

Any of the following can be supplied in any order.

Name	Description
`<Format>`	Content is name identifier format which is added to all the applicable roles of the entities which match any of the following `<Entity>` or `<ConditionRef>` elements.
`<Entity>`	The textual content is an EntityID. All preceding formats are added to applicable roles of the entity with this ID.
`<ConditionRef>`	The textual content is the Bean ID of a Predicate<EntityDescriptor>. All preceding formats are added to the roles of the entities for which this returns true.
`<ConditionScript>` 3.4	The content of this element is an inline or local script resource that implements Predicate<EntityDescriptor>. All preceding formats are added to the entities for which this returns true.

Examples

The example will add the "persistent" format to the first entity, and both the "persistent" and "email" formats to the second.

Add NameIDFormat elements to metadata

```
<MetadataFilter xsi:type="NameIDFormat">
        <Format>urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</Format>
        <Entity>https://sp1.example.org</Entity>
        <Format>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</Format>
        <Entity>https://sp2.example.org</Entity>
</MetadataFilter>
```

The following example using new features specific to V3.4 is similar, but the specification of the entities to apply the formats to is handled with inline scripts. Obviously these scripts aren't particularly useful but they demonstrate the syntax.

V3.4+: Use of scripts

```
<MetadataFilter xsi:type="NameIDFormat">
        <Format>urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</Format>
        <ConditionScript>
            <Script>
            <![CDATA[
                    input.getEntityID().equals("https://sp1.example.org");
            ]]>
            </Script>
        </ConditionScript>
        <Format>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</Format>
        <ConditionScript>
            <Script>
            <![CDATA[
                    input.getEntityID().equals("https://sp2.example.org");
            ]]>
            </Script>
        </ConditionScript>
</MetadataFilter>
```