SPAttributeConfig

Shibboleth allows a user and a site to release a varying set of attributes to a destination site, and does not impose restrictions on the kinds of attribute information provided by an AA. SP implementations must be prepared to examine the attributes they receive and filter them based on policies about what information to permit an IdP to assert about its users.

Attribute acceptance is the process of defining acceptable attributes and filtering attribute values before passing them on to a resource manager, such as the Shibboleth module or a web application. Data blocked by AAP filters will not be passed to the CGI environment or used when enforcing `.htaccess` rules in Apache. Note that the attribute assertion exported to the `HTTP_SHIB_ATTRIBUTES` header is now also filtered. This is a change from previous versions. To compensate, either no AAP can be specified, or a rule can be applied to permit all attributes to pass through while also exporting specific attributes.

An essential part of the Shibboleth trust fabric is ensuring that sites only assert attributes for domains for which they are considered authoritative by the SP. These domains are defined in the metadata associated with each IdP using `<shibmd:Scope>` elements. Typically, Brown University would only be trusted to assert attributes only scoped to `brown.edu`. Some deployment scenarios may exist where Brown would be authoritative for satellite organizations, which will be reflected in the metadata. Unless there are very specific circumstances requiring this restriction be removed, it is strongly encouraged that such policies be enforced.

By default, Shibboleth releases the entire set of attributes specified for release to a relying party in any given transaction. To request a smaller set of specific attributes or none at all, `<saml:AttributeDesignator/>` elements may be added to `<Application>` elements. This is primarily useful to restrict the information exported to specific applications on a web server that are less trusted.

```
<saml:AttributeDesignator AttributeName="_name_" AttributeNamespace="_namespace_"/>
```

`AttributeDesignator`	is used in the `Applications` and `Application` elements to name an attribute to specifically request from IdPs on behalf of an application. If this element is not present, the application will be given anything the IdP allows it to receive.
`AttributeName`	Specifies the name of a SAML attribute, generally a URI.
`AttributeNamespace`	Specifies the attribute's SAML namespace, which Shibboleth by convention sets to `urn:mace:shibboleth:1.0:attributeNamespace:uri`.

The default set of designators can be overridden within individual `Application` elements, but if `AttributeDesignator` elements are specified in the main `Applications` element, it isn't possible to "remove" them and revert to none within a particular application.

Attribute Acceptance Policies (AAPs)

The Shibboleth implementation supports Scoped and Simple attributes and filtering policies for different kinds of attributes, and is potentially extensible to more complex attributes in the future. An attribute is considered Scoped if the XML representation of its values contains a "Scope" attribute. This is detected at runtime and requires no configuration in advance. **It is strongly recommended that complex XML structures for attributes be avoided at all costs due to difficulties in interoperability and processing.**

NameIdentifiers are supplied by the IdP in the subject of SAML assertions to identify the type of user identifier that has been passed. In some cases, this information may be useful for applications and access control decisions. These values can be placed in headers using AAPs by defining an `AttributeRule` with a `name` matching the `format` of the identifier asserted by the IdP. Handles (`urn:mace:shibboleth:1.0:nameIdentifier`) should never be used for access control decisions.

Scoped Attributes

Scoped attributes are a special kind of attribute whose values are a combination of a `value` and a `scope`, or `context` for the value. An example is `eduPersonScopedAffiliation`, which adds a scope to the defined set of `eduPersonAffiliation` values, such as `student`, `member`, or `faculty`. Scopes are expressed as DNS domains and subdomains as a convention.

Any `scoped` attribute can be scoped only to the IdP's permitted domains as defined by `<shibmd:Scope>` elements in that IdP's metadata. This policy information can be overridden or supplemented using the AAP. Domains can be explicit or regular expressions, and can be changed by a SP to meet its needs. Thus, attribute acceptance processing for `scoped` attributes is based on site metadata and SP-specified overrides in addition to the mechanism described below for `simple` attributes.

Scope rules specified in an AAP are additive with any domains permitted by site metadata, and the rules are applied by first looking for an applicable denial rule, and then looking at site metadata and any applicable site rules for an accept rule.

Simple Attributes

Simple attributes are attributes whose value is expressed in XML as a Text node; that is, the value is just a string. Multiple values are permitted. `eduPersonEntitlement`, in which the values are URIs, is one example of a simple attribute.

`AAP.xml`

Both Simple and Scoped attribute acceptance is controlled with an external (or in 1.2, optionally inline) policy file written in XML. The schema for the file is described by the `shibboleth.xsd` schema, and an example file is included, `AAP.xml`. It is now optional to supply such a policy, but in the absence of one, no attributes will be exported into request headers, and the option to export the assertion as a whole must be used instead.

The policy is a default-deny algorithm that requires permissible attributes and values be listed explicitly. That is, an empty (as opposed to no) policy permits nothing. Each attribute to be supported must be listed in the policy by name in an `<AttributeRule>` . Each such rule is a collection of `<SiteRule>` elements along with an optional `<AnySite>` default rule. In turn each site rule is a set of `<Value>` rules that specify matches to permit, either literal or regular expressions, or a wildcarded `<AnyValue>` default rule, which is equivalent to a single regular expression rule allowing anything.

`<AnyAttribute>` elements can be used before or in place of the `<AttributeRule>` elements to allow all attributes and values to be accepted. The purpose of this is to then supply rules to specify the export of particular attributes, without using those rules to control acceptance. Be careful when using this directive.

Attribute rules are written by individual elements identified by attribute name, then filtered by providers to which the rule applies. Further filtering can be applied by value to impose a controlled vocabulary. A simple rule looks like:

```
<AttributeRule Name="urn:mace:dir:attribute-def:eduPersonPrincipalName" Scoped="true" Header="REMOTE_USER"
Alias="user">
        <AnySite>
                <Value Type="regexp">^[^@]+$</Value>
        </AnySite>
</AttributeRule>
```

Attribute Rule Syntax

```
<AttributeRule
         Name="urn:mace:dir:attribute-def:eduPersonScopedAffiliation"
         Header="Shib-EP-Affiliation" Alias="affiliation">
```

`Name`	The name of the Shibboleth attribute, usually a URI. This is the only required XML attribute.
`Namespace`	If the attribute's name includes a SAML namespace, supply it here. Normally this is unused.
`Header`	The name of a HTTP request header to map the value of this attribute into.
`Alias`	A short name for the attribute to be used as a reference for this rule by Apache `Require` directives.

`<AttributeRule>` elements will either contain an `<AnySite>` element or one or more `<SiteRule>` elements defining to which IdP's this rule will be applied.

`<AnySite>`	Specifies a rule that always applies to the attribute, regardless of the asserting IdP.
`<SiteRule Name="providerId">`	A rule that only applies to the IdP or federation corresponding to the supplied `providerId`.

Individual `<SiteRule>` elements can further contain limitations on the attribute values or scopes that that providerId is allowed to express. This can be useful for controlled vocabularies, to expand or constrain the domains for which an IdP is authoritative, or to control the entitlements accepted from a particular site.

`<AnyValue>`	Specifies a rule that always applies to the attribute and site, regardless of the value(s).
`<Value Type="type">`*`string`*`</Value>`	Specifies a value to permit, either directly using type `literal`, or using a set of matching expressions as type `regexp`. `literal` is the default if type is not specified.
`<Scope Accept="true /false" Type="type">`	Specifies a value to accept or deny, either directly using type `literal`, or using a set of matching expressions as type `regexp`. `literal` is the default if type is not specified. Accept defaults to "true".

The regular expression syntax is a subset of the usual Perl and Unix syntaxes that is described in the XML Schema specification by the W3C. Most typical expressions should work. Be sure to anchor them using ^ and $ to avoid unintentional matches midstring.

The operative AAP engine and `AAP.xml` configuration file are specified in `shibboleth.xml` using the `<AAPProvider>` element, and default to `type="edu.internet2.middleware.shibboleth.aap.provider.XMLAAP"` and `uri="/opt/shibboleth-sp/etc/shibboleth/AAP.xml"`.