AttributeAcceptancePolicy

Summary

An AAP is a policy defining rules for the processing of SAML attributes by a ServiceProvider. These rules include:

- Whether to "accept" the attribute's values. Unaccepted values are filtered out before information is extracted for use by applications. The raw, unfiltered assertions can still be accessed when required.
- The HTTP request headers in which to place attribute values. Multiple attributes can be mapped to a single header. Multiple values are collectively combined into a single header with semicolons separating the values.
- Aliases by which attributes can be referenced in access control policy rules, such as Apache require commands. SAML attributes tend to have machine-friendly names, so aliases allow more admin-friendly names to be used instead.

News

The ShibOnedotThree ServiceProvider includes a backward-compatible, but revamped system for AAP processing. Roughly the same system should be included in the Java ServiceProvider. New features include:

- Case-insensitive value filtering, reducing the need for complex regular expressions.
- Elimination of ineffective and inconsistent syntax.
- Support for combining multiple policies.
- Ability to supply attribute definitions separately from extraction rules, enabling centralized distribution of policies.

Details

The <`AnyAttribute`> wildcard is a signal in a policy file to bypass any filtering at all, and only use the AAP mechanism to configure header export and alias rules. This enables filtering rules to be supplied in one file, but specific export rules to be supplied in a second file. Without this signal, processing proceeds as follows for each SAML attribute received:

> *Each policy is examined for a matching <`AttributeRule`> and if none are found, then the attribute is ignored (accepted) for the purposes of that policy. If more than one policy contains a matching rule, then all are applied conjunctively (a value must be acceptable to all policies for it to pass through unmolested). At the end of the process, if no policy contains a rule for a particular attribute, it is rejected/deleted.*
>
> *In each matching <`AttributeRule`> rule, an empty rule means to accept any values. Otherwise, the <`AnySite`> and any matching <`SiteRule`> elements are evaluated. Matching is based on the `entityID` of the IdentityProvider but also on any containing named <`EntitiesDescriptor`> elements. Rules are evaluated in sequence from most specific to least specific match.*
>
> - *An empty <`SiteRule`> (or =<AnySite>=) element blocks all values. Once a site rule applies, the default policy becomes* bl ock anything that is not specifically permitted*.*
>
> - *The <AnyVa lue> element is an* accept all *policy for the matching rule. Otherwise, <Va lue> elements are applied in sequence to determine whether each value asserted is acceptable under the policy:*
>   - *Any <Va lue> elements with Accept {{ =false}} take effect.*
>   - *Any <Va lue> elements with Accept {{ =true}} (the default) take effect.*

Scoping

The Shibboleth implementation includes support for so-called *scoped* attributes, which are relations between a value and a *scope* or *context* for the value. This enables the scope to be processed separately from the value instead of treating both as a combined string to be filtered.

Scope filtering happens along with, but distinctly from, value filtering rules. Within a given policy, scope filtering runs whenever a matching <`AttributeRule`> contains a `Scoped` XML attribute set to `true` . This means that a single policy indicating an attribute is scoped will ensure that (in SAML 1.x assertions) each <`saml:AttributeValue`> contains a `Scope` XML attribute, preventing any attempts by an IdentityProvider to subvert the rules. Even if SP-specific policies were to omit the flag, a single federation-supplied "definition" file is enough to get the basics correct. The specifics of this will change for SAML 2.0 assertions, but the rules will still be enforced properly.

Any time scope filtering is in effect for an <`AttributeRule`> , the default rule in effect is that any scope is illegal. The rule then denies or accepts specific scopes for specific IdentityProviders. The <`AnySite`> and any matching <`SiteRule`> elements are examined for scope rules. As above, matching is based on the `entityID` of the IdentityProvider and any containing named <`EntitiesDescriptor`> elements, and rules are evaluated most-specific to least-specific.

- Any <`Scope`> elements with `Accept=false` take effect.
- Any <`Scope`> elements with `Accept=true` (the default) take effect.

Finally, any <`shibmd:Scope`> extension elements in MetaData take effect (these are always used to indicate acceptance).

Examples

Accept eduPersonPrincipalName in scope `example.org` .

```
<AttributeRule Name="urn:mace:dir:attribute-def:eduPersonPrincipalName"
                                         Scoped="true"
                                         Header="REMOTE_USER"
                                         Alias="user">
         <AnySite>
                 <Scope Accept="true">example.org</Scope>
                 <AnyValue/>
         </AnySite>
 </AttributeRule>
```