IdPXMLSigEnc

Configuring XML Signature and Encryption

Settings for XML signing and encryption are always configured on a per relying party, or relying party group, basis within the *$IDP_HOME/conf/relying-party.xml* file.

The IdP is capable of determining if the message integrity protection, provided by XML signatures, and the message confidentiality protections, provided by XML encryption, are provided by the communication channel and message encoding style used to respond to messages. As XML signing and encryption are significantly more time and resource intensive, when compared to transport/encoding level mechanisms, the IdP allows deployers to indicate whether these XML operations are always required, required only if the transport/encoding level doesn't provide them, or never required. These options are represented, respectively, by the values *always*, *conditional*, and *never* used in the attributes that control whether some portion of XML data is signed or encrypted. These attributes are described below.

XML Signature Configuration

Defining the Signature Credential

The X.509 credential that is used to sign SAML messages may be configured in two places:

- the attribute `defaultSigningCredentialRef` located on a `RelyingParty` element
- the attribute `signingCredentialRef` located on a `ProfileConfiguration` element
  As you may have guessed, the `defaultSigningCredentialRef` attribute is used to specify a default signing credential for every profile configuration contained with the `RelyingParty` while the `signingCredentialRef` attribute is used to specify a signing credential only for the `ProfileConfiguration` upon which it appears and it always overrides a default signing credential if one is specified. The value for both attributes is the ID of a credential defined within the *$IDP_HOME/conf/relying-party.xml* file. **Note:** the referenced credentials MUST contain a private key as this is what is actually used to sign the XML.

```
<RelyingParty id="urn:example.org" provider="http://idp.example.org" defaultSigningCredentialRef="
ExampleOrgCred">
    <ProfileConfiguration xsi:type="saml:ShibbolethSSOProfile" />
    <ProfileConfiguration xsi:type="saml:SAML1AttributeQueryProfile" signingCredentialRef="SAML1AACred"/>
    <ProfileConfiguration xsi:type="saml:SAML1ArtifactResolutionProfile" />
    <ProfileConfiguration xsi:type="saml:SAML2SSOProfile" />
    <ProfileConfiguration xsi:type="saml:SAML2AttributeQueryProfile" />
    <ProfileConfiguration xsi:type="saml:SAML2ArtifactResolutionProfile" />
</RelyingParty>
```

In the above example we define a default signing credential called "ExampleOrgCrd" and then define another credential, used only for SAML 1 attribute queries, called "SAML1AACred". This means that when the IdP is responding to a SAML 1 attribute query it will use the "SAML1AACred" credential for signing messages but when it responds to any other profile request it will use the "ExampleOrgCred" credential.

Controlling what is Signed

Once you defined the signing credential to be used by the IdP you can then configure what the IdP will sign. This is controlled by three, boolean, attributes located on a `ProfileConfiguration`:

- **signResponses** - indicates whether response messages should be signed, permissible values: *always*, *conditional*, *never* (default value: conditional)
- **signAssertions** - indicates whether assertions, within a message, should be signed, permissible values: *always*, *conditional*, *never* (default value: never)
- **signRequests** - indicated whether request message should be signed, permissible values: *always*, *conditional*, *never* (default value: conditional). This option is not currently used by the IdP because there is currently no support for profiles that require the IdP to make a request to another party. The first such profile will likely be Single Logout.

```
<RelyingParty id="urn:example.org" provider="http://idp.example.org" defaultSigningCredentialRef="
ExampleOrgCred">
    <ProfileConfiguration xsi:type="saml:ShibbolethSSOProfile" />
    <ProfileConfiguration xsi:type="saml:SAML1AttributeQueryProfile"/>
    <ProfileConfiguration xsi:type="saml:SAML1ArtifactResolutionProfile" />
    <ProfileConfiguration xsi:type="saml:SAML2SSOProfile" signAssertions="always"/>
    <ProfileConfiguration xsi:type="saml:SAML2AttributeQueryProfile"/>
    <ProfileConfiguration xsi:type="saml:SAML2ArtifactResolutionProfile" />
</RelyingParty>
```

In this example we override the default value for assertion signing on SAML 2 SSO requests, for example, because the relying party may wish to remove those assertions from the incoming SAML response, use them within another message, but still be able to validate the integrity of the assertions.

XML Encryption Configuration

Defining the Encryption Credential

Unlike the with XML signature support the IdP does not define the credential it will use for encryption, the relying party does. This means, in order to support encryption, the relying party must publish its public key within its metadata. This key must have either no `use` attribute (indicating the key should be used for signing or encryption) or have a `use` attribute with a value of "encryption" indicating the key should only be used for encryption.

Controlling what is Encrypted

First, only SAML 2 supports encryption so encryption can not be enabled on SAML 1 profile. Second, the IdP currently only supports the encryption of assertions and NameIDs, it does not support the encryption of attributes (though this will be added in the near future).

Encryption of assertions and NameIDs is controlled by two, boolean, attributes located on a SAML 2 `ProfileConfiguration`:

- **encryptAssertions** - indicates whether assertions should be encrypted, permissible values: *always*, *conditional*, *never* (default value: conditional)
- **encryptNameIds** - indicates whether the NameIDs should be encrypted, permissible values: *always*, *conditional*, *never* (Default value: never)

```
<RelyingParty id="urn:example.org" provider="http://idp.example.org" defaultSigningCredentialRef="
ExampleOrgCred">
    <ProfileConfiguration xsi:type="saml:ShibbolethSSOProfile" />
    <ProfileConfiguration xsi:type="saml:SAML1AttributeQueryProfile"/>
    <ProfileConfiguration xsi:type="saml:SAML1ArtifactResolutionProfile" />
    <ProfileConfiguration xsi:type="saml:SAML2SSOProfile" encryptNameIds="conditional" />
    <ProfileConfiguration xsi:type="saml:SAML2AttributeQueryProfile" />
    <ProfileConfiguration xsi:type="saml:SAML2ArtifactResolutionProfile" />
</RelyingParty>
```

In this example we override the default for encrypting NameIDs.