SAML1AttributeQueryConfiguration

File(s): *conf/relying-party.xml*

Format: Native Spring / Deprecated Custom Schema

Legacy V2 File(s): *conf/relying-party.xml*

Overview

The **SAML1.AttributeQuery** profile configuration bean enables support for the SAML 1.1 Attribute Query profile over SOAP. It was historically used to support Shibboleth SP software that understood how to supplement SAML 1.1 SSO with queries in order to improve attribute confidentiality. It is rarely needed any longer.

By default, the IdP will only respond to queries containing NameIdentifiers that it understands how to reverse-map into user identities, and it will not do so out of the box for anything but transient identifiers issued by it. Nevertheless, it should be disabled if not in use.

Configuration

The most typical options used are described in more detail below, but not every obscure option is discussed. See the javadoc for all of the possible configuration options for this profile (note that many of them are inherited from parent classes).

Virtually all the configuration options below can be set via two different properties: a static property that explicitly sets the value to use and a lookup strategy or predicate property that takes a Function or Predicate and returns the value to use. The dynamic property is generally named "propertyNamePredicate" or "propertyNameLookupStrategy" for Boolean- and non-Boolean-valued properties respectively.

The examples shown are not specific to any particular profile configuration.

Common

Options common to most/all profiles:

Name	Type	Default	Description
securityConfiguration	SecurityConfiguration	Bean named **shibboleth. DefaultSecurityConfiguration**	An object containing all of the default security-related objects needed for peer authentication and encryption. See SecurityConfiguration for complete details.
disallowedFeatures [3.3]	Integer	0	A bitmask of features to disallow, the mask values being specific to individual profiles

Guidance

Modifying the security configuration is usually done to:

- specify an alternate signing or decryption key to use
- control signing or encryption algorithms (but for metadata you control, it's advisable to control algorithms by using an extension to specify supported algorithms).

SAML

Options common to SAML profiles:

Name	Type	Default	Description
additionalAudiencesForAssertion	Collection<String>		Additional values to populate into audience restriction condition of assertions
includeConditionsNotBefore	Boolean	true	Whether to include a `NotBefore` attribute in assertions
assertionLifetime	Duration	PT5M	Lifetime of assertions
signAssertions	Predicate<ProfileRequestContext>	false	Whether to sign assertions
signResponses	Predicate<ProfileRequestContext>	varies by profile	Whether to sign responses
signRequests	Predicate<ProfileRequestContext>	false	Whether to sign requests

Guidance

It isn't too common to need any of these options, and they should be changed only with care.

The `additionalAudiencesForAssertion` and `includeConditionsNotBefore` settings provide ways to work around bugs in other systems. You should never use these settings without obtaining a commitment from the other system's owner to fix their bugs.

The `assertionLifetime` setting does **not** involve control over the session with the relying party, it's only relevant in delegation scenarios.

The signing options have a complex history, which is one reason they are not themselves just boolean-valued. We provide Spring support so you can just set them to "true" or false" as though they are, but they also directly support the more dynamic approach of deriving the value with a bean.

The `signResponses` default varies by profile, see the notes on the individual profile pages.

If you need to enable the `signAssertions` option, and you control the SP's metadata, you should generally add the `WantAssertionsSigned` flag to it in place of using this option.

Notes

The default value of `signResponses` for this profile is an extended form of the behavior that was referred to in V2 as "conditional". It signs only if TLS isn't used (very unusual) or if the receiving port is 443. It assumes that traffic over 443 will be relying on message-based security measures (but see below), whereas traffic to an alternative TLS port like 8443 will be relying on mutual authentication and thus provide a secure channel.

Since SAML 1.1 does not support XML Encryption, all data is in plaintext, and therefore use of message-based security is not advisable.