RelyingParty

The <RelyingParty> element allows the SP to customize its behavior when it interacts with particular identity providers or groups of providers. By default, many properties are set globally to an application (or usually the whole system). This element allows individual options to be selectively overriden.

Selection of when to apply the override via the element can be entityID-based, group-based, or can rely on an extensible matching mechanism called an EntityMatcher.

If the Name attribute is present, then the matching process starts with the IdP's entityID, and proceeds upwards through the IdP's Metadata matching against <EntitiesDescriptor> group names that are found. The most specific match wins.

If the Name attribute is not present, then a type attribute must be used to indicate the type of EntityMatcher to apply, and other content will be required based on the type of matcher to specify how to match.

Attributes

Name	Type	Default	Description
Name	string		For name-based matching, this is the value used to match against the IdP's entityID or parent group names. This attribute is optional and can be omitted in favor of a type attribute.
type	string		For extensible matching, specifies the type of EntityMatcher to use. Refer to the associated documentation for additional required content.
entityID	URI		Overrides the unique identifier used by the SP to identify itself when communicating with matching relying parties. Normally an SP should be able to use a single name in all its dealings, but this can provide some help when dealing with externally imposed limitations. Again, this is not the IdP's name, but the SP's name.
authType	See Description	TLS	Specifies the transport-layer authentication mechanism that is used for back-channel SOAP messages to an IdP. The values permitted are implementation dependent, but may include: **TLS** client certificate TLS/SSL authentication **basic** HTTP Basic-Auth (cleartext name/password) **digest** HTTP Digest-Auth **ntlm** Microsoft's NTLM authentication **gss** GSS-API (SPNEGO)
authUsername	string		Required for non-TLS and GSS authType values, this is the username to use
authPassword	string		Required for non-TLS and GSS authType values, this is the password to use
signing			Controls outbound signing of XML messages. . See Signing&Encryption
signingAlg	URI	*specifier for* RSA-SHA1	An XML Signature signature algorithm specifier for signatures produced by the SP.
digestAlg	URI	*specifier for* SHA1	An XML Signature digest algorithm specifier for signatures produced by the SP.
encryption			Controls outbound encryption of XML messages and content. See Signing&Encryption
encryptionAlg	URI	*specifier for* RSA-OAEP-SHA1	An XML Encryption key wrap/transport algorithm specifier for encryption performed by the SP. The actual symmetric encryption algorithm will be derived from it.
keyName	string		Specifies a particular credential to use for signing or TLS authentication by attaching a name to the lookup criteria passed to the credential resolver in use. Typically the credential resolver will be able to attach names or aliases to credentials in some fashion. For more on using this feature, see the Multiple Credentials topic.
artifactEndpointIndex	string		Identifies which <ArtifactResolutionService> handler at the SP is used when sending artifact-bound messages to the relying party. Endpoints typically include an index attribute to copy here.
chunkedEncoding	boolean	false	Controls the use of chunked encoding during back-channel SOAP communication. HTTP clients sending data must either compute and send a Content-Length header to the server (requiring that all data be buffered ahead of time), or use chunked encoding. A lot of servers mis-handle this option, so it is disabled by default.
connectTimeout	time in seconds	10	Specifies the timeout for connecting to remote servers during back-channel SOAP communication.
timeout	time in seconds	20	Specifies the total time to allow for completing back-channel SOAP communication.

`requireConfidentiality`	boolean	true	When true, the SP will require the use of TLS/SSL for all back-channel SOAP communication. This prevents an unsafe exchange of data **before** an unencrypted channel might be used, since XML encryption depends on the peer's willingness to use it.
`requireSignedAssertions`	boolean	false	When true, assertions **MUST** be digitally signed, regardless of any other signatures used to authenticate them. Typically needed only for advanced auditing or assertion forwarding use cases.
`requireTransportAuth`	boolean	true but look here	When true, the SP will require back-channel SOAP communication to be authenticated at the transport layer (TLS/SSL server authentication). See the this topic for additional semantics
`sessionHook`	URL		Specifies a location to send the client after a session has been created (i.e., after login), but before transferring the client to the eventual final resource. This is normally a relative path to ensure that the session will be visible to the hook script, but doesn't have to be. A hook can be used to validate something about the session to check its "fitness for purpose" before delivering the client to an application that may not offer sufficient error handling capability to do the job itself. A common example is checking for required attributes. The hook redirect will include two parameters, `target` and `return`. The `target` parameter contains the resource URL that will eventually be the client's destination, in case the hook cares. The `return` parameter is the location to redirect the client back to upon completion of the hook. The hook **MUST** either redirect back or take complete ownership of the client with no further processing by the SP.
`artifactByFilesystem`	boolean	false	Enables the artifact-based "back-door" external authentication mechanism described in the BackDoor topic.
`cipherSuites`	OpenSSL cipher expression	see description	Directly configures the SSL/TLS ciphers to support when making SOAP connections. The default value (`ALL:!aNULL:!LOW:!EXPORT:!RC4:!SSLv2`) is historical and has been in place for a few releases, and has been left alone to prevent upgrades from affecting interoperability. A stronger value is now used in the default files distributed with the software, which was derived from Mozilla's tool. Note that this does not include or affect TLS 1.3 ciphers.
`authnContextClassRef`	space delimited URIs		Supplies values for the SAML 2.0 `<AuthnContextClassRef>` element in requests to applicable IdPs, or for the `wauth` parameter in WS-Federation requests. Ignored for other protocols.
`authnContextComparison`	"exact", "minimum", "maximum", "better"		Supplies values for the `<RequestedAuthenticationContext>` comparison operator in SAML 2.0 requests to applicable IdPs. Ignored for other protocols.
`NameIDFormat`	URI		Supplies a value for the `<NameIDPolicy>` element's `Format` attribute in SAML 2.0 requests to applicable IdPs. Ignored for other protocols.
`SPNameQualifier`	URI		Supplies a value for the `<NameIDPolicy>` element's `SPNameQualifier` attribute in SAML 2.0 requests to applicable IdPs. Ignored for other protocols

Example

The example demonstrates requesting a different format of `<NameID>` from a particular IdP.

```
<ApplicationDefaults ... >

    ...

    <RelyingParty Name="https://idp.example.org/idp/shibboleth"
                NameIDFormat="urn:oasis:names:tc:SAML:2.0:nameid-format:persistent"/>

    ...

</ApplicationDefaults>
```