# IdP Change Proposals



## 1 Delegate Token Creation Policy

The following policy points control the creation of a delegate token. They do not effect the content of the issued assertion.

### 1.1 SPs Allowed to Request a Delegate Assertion

This policy decision controls which service provider may request a delegate SAML token based on a SAML token it has previously obtained.

This policy setting will be exposed by means of a new attribute located on a `<RelyingParty>` configuration element. The attribute, `allowTokenDelegation`, is a boolean. The default value is false.

**Example RelyingParty configuration**

```
<RelyingParty id="http://example.org/shibboleth"
              provider="http://example.org/idp/shibboleth"
              allowTokenDelegation="true" />
```

### 1.2 Maximum Delegation Chain Limit

This policy decision limits the total number of delegates that may be derived from the initial SAML token. For example, if the limit is 2 then SP A may delegate to SP B (1 delegation), SP B may then delegate to SP C (2 delegations), but SP C may not delegate to anyone else.

This policy setting will be exposed by means of a new attribute located on a `<RelyingParty>` configuration element. The attribute, `maximumTokenDelegationChainLength`, will be a positive integer. The default value will be 1. Note, while this attribute may appear on a `<RelyingParty>` configuration to which a SAML token is delegated, it is only configuration for the SP to which the token was initially issued that controls the chain length. Obviously this value is meaningless if `allowTokenDelegation` is set to false.

**Example RelyingParty configuration**

```
<RelyingParty id="http://example.org/shibboleth"
              provider="http://example.org/idp/shibboleth"
              allowTokenDelegation="true"
              maximumTokenDelegationChainLength="5" />
```

### 1.3 Lifetime of a Delegated Assertion

This policy setting controls the lifetime of a delegate SAML token; that is it corresponds to the lifetime of the delegate assertion issued to the service provider.

This policy setting will be exposed by means of a new attribute located on a `<RelyingParty>` configuration element. The attribute, `delegateTokenLifetime`, will be duration. The default value will be 8 hours. It is the value associated with the requesting service provider that is used for this setting. Therefore this setting, on the service provider requesting the initial delegate token.

**Example RelyingParty configuration**

```
<RelyingParty id="http://example.org/shibboleth"
              provider="http://example.org/idp/shibboleth"
              allowTokenDelegation="true"
              maximumTokenDelegationChainLength="5"
              delegateTokenLifetime="P24H" />
```

## 1.4 SPs to which an Assertion may be Delegated

This policy decision controls to which service providers a delegate token may be issued.

This policy setting will be exposed by a new child element, `<DelegationRestriction>`, located within the SAML 2 SSO profile configuration element. The contents of the element will be the entity ID to which the assertion may be delegated. The element may appear multiple times to indicate that the delegate may be issued to any one of the specified service providers.

**Example SAML 2 SSO Profile Configuration with Delegation Restriction**

```
<ProfileConfiguration xsi:type="saml:SAML2SSOProfile">
    <DelegationRestriction>http://sp2.example.org/shibboleth</DelegationRestriction>
    <DelegationRestriction>http://sp3.example.org/shibboleth</DelegationRestriction>
    <DelegationRestriction>http://sp4.example.org/shibboleth</DelegationRestriction>
</ProfileConfiguration>
```

# 2 Delegate Token Content

The following sections describe the content within a delegate token.

## 2.1 Name Identifiers and Attributes

No changes to the current IdP attribute creation and filtering mechanism is currently proposed. The current filtering mechanism, which of most interest in this case already supports the release of attributes on a per-SP basis. Additionally, the filtering engine already has support for blocking the release of attributes if a user is not logged in, functionality that was identified as useful in this use case.

## 2.2 Expression of Delegation Chain

Current practice in this area is limited to advisory information that cannot prevent acceptance of an assertion by current delegation-ignorant implementations, including non-Shibboleth SP software. Preventing this was identified as an important requirement, therefore a new type of SAML condition has to be created. Since a new condition is needed anyway, it seems better to consolidate the ability to identify the delegation chain in one place, so that both the number of intermediaries and their identities can be obtained and controlled using one structure.

A draft proposal has been submitted to OASIS:

http://wiki.oasis-open.org/security/SAML2DelegationCondition

We would ignore the ID-WSF-defined `<TransitedProviders>` advisory approach, and rely on something like this:

**Example DelegationRestriction Condition**

```
<saml:Condition xsi:type="del:DelegationRestrictionType">
    <del:Delegate>
        <saml:NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">
          https://portal.example.edu/shibboleth
        </saml:NameID>
    </del:Delegate>
    <del:Delegate>
        <saml:NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">
          https://portal.example.edu/portlet1/shibboleth
        </saml:NameID>
    </del:Delegate>
</saml:Condition>
```

The order is least to most recent intermediary, and the last value would also appear in the `<saml:SubjectConfirmation>` element for consistency with the original standard.

Additional attributes on the `<del:Delegate>` element would be optionally available to identify the time of delegation and the confirmation method used (as in ID-WSF).

Whatever the final syntax, the SP would then require explicit modification to recognize this new condition type and enforce any policy on what shows up.

## 2.3 Audience Restriction

Currently the IdP adds an audience restriction listing the service provider for whom the assertion is issued. Tokens that may be delegated will also carry the entity ID of the IdP as an audience of the assertion.

# 3 Security Policies

## 3.1 Inbound Request Security Policy

The existing IdP security policies cover most of the checks necessary for the validation of an incoming delegation request. These includes replay attacks, issue instant checking (as controlled by the setting in section 1.3), and Service Provider identification (through either client-cert auth or XML DSig).

The only additional check that seem necessary is service provider presenting the token to be delegated is the service provider to which that token was issued. That is, that its entity ID matches the last ID in the delegation chained as identified using the semantics given in section 2.2. There a new security policy rule, {{ }}, will be created and added to the default security policy for the SAML 2 SSO endpoint. As with other rules, this rule will pass on the incoming request if no delegation restriction is present in the SAML request.

## 3.2 Outbound Response Security

The default configuration the IdP does not sign the SAML `Assertion` contained in a `Response`. Service Providers that are marked with the ability to request delegate tokens, per section 1.1, will automatically have their configuration changed to issue signed assertions, effectively making the `signAssertions` attribute on the `SAML2SSOProfile` profile configuration default to the value "always". If this attribute is explicitly set within the configuration then it will be honored.

## 3.3 SAML Token validation

*This is modeled after the current SP token validation.*

The security token will be validated by running it through a `SecurityPolicy` with the following set of `SecrityPolicyRule` rules:

- Assertion "freshness" validation based on IssueInstant and setting in section 1.3
- Signature validation
- `Conditions` validation. This rule would take a set of condition validation plugins. Initial implementations would be provided for `AudienceRestriction` and `DelegationRestrictionType` conditions. The `Conditions` rule would verify the `NotBefore` and `NotOnOrAfter` attributes as well as ensure all the `Condition` elements were processed.
- Subject confirmation validation. This rule would be populated with a set of confirmation method plugins. Initial plugins will be provided for bearer and holder-of-key methods.

This is the same mechanism used by the IdP to evaluate the security of incoming messages as well. It should prove to be very flexible in supporting new checks, new conditions, and new subject confirmation methods.

## 3.4 User Authentication via SAML Token

*In Progress*

1. Extract the SAML token from the WS-Security header
2. Validate it via mechanism listed in section 3.3
3. Extract NameID (encrypted NameID will not be supported) and map to a principal via a principal connector

Do we need to worry about attributes or can we simply requires the NameID must be mappable to the principal name.