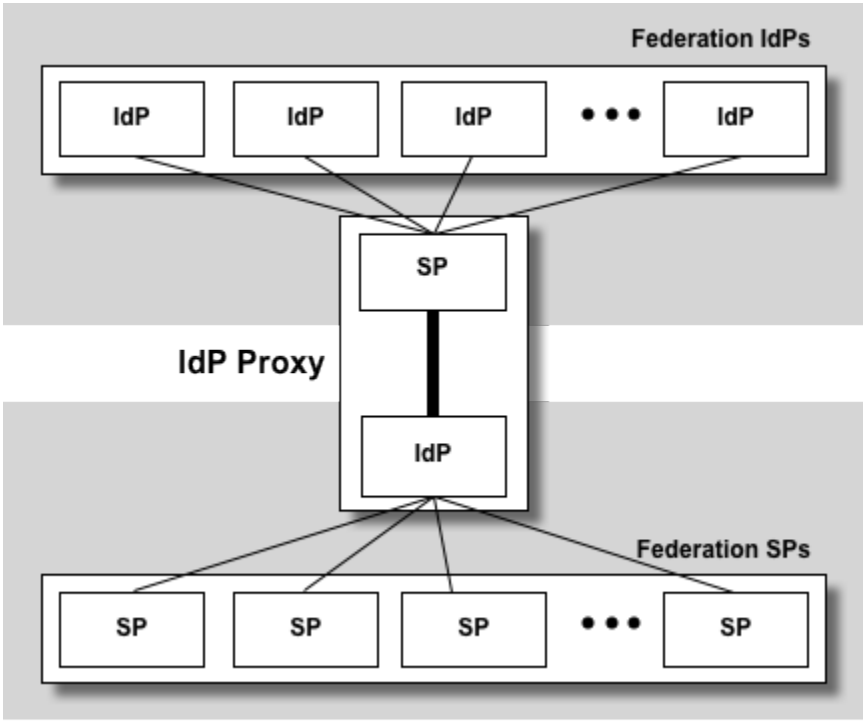
SAMLIdPProxy

SAML IdP Proxy

A *SAML IdP Proxy* is a bridge or gateway between a federation of SAML IdPs and a federation of SAML SPs:

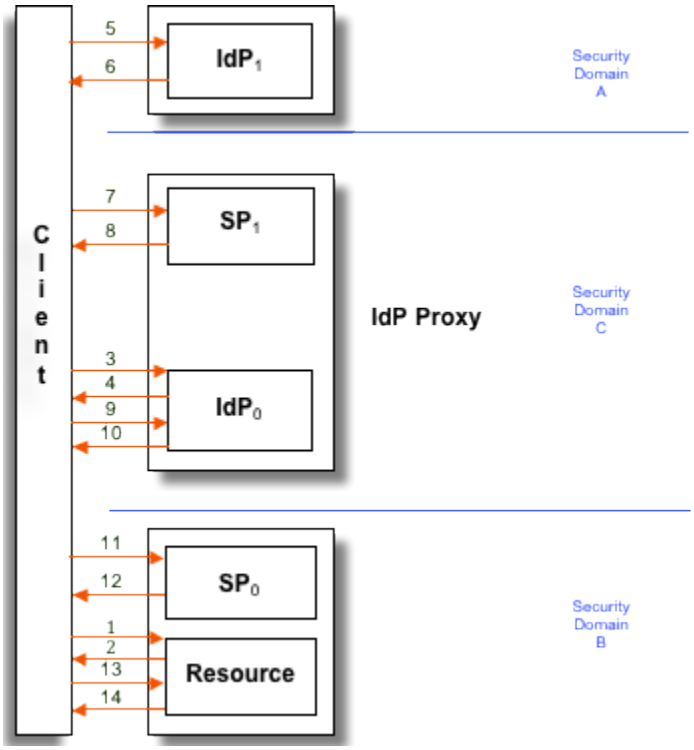


To an SP, an IdP Proxy looks like an ordinary IdP. Likewise, to an IdP, an IdP Proxy looks like an SP. Thus an IdP Proxy has the combined capability of both an IdP and SP.

Like a Web (HTTP) Proxy, an IdP Proxy delivers increased efficiency, security, and flexibility.

	Web Proxy	**IdP Proxy**
Efficiency	cache web pages	cache attributes
Security	controlled access to web pages	controlled access to federation IdPs
Flexibility	HTTP request/response filtering	SAML request/response filtering

The following flow diagram illustrates the relationship among the various IdP Proxy components:



Here is an outline of the typical IdP Proxy flow:

1. A browser client requests a web resource protected by a SAML SP (SP_0). If a security context for the principal already exists at SP_0, skip to step 14.
2. The client is redirected to the IdP component of the IdP Proxy (IdP_0), which is protected by the SP component of the IdP Proxy (SP_1).
3. The client makes a SAML `AuthnRequest` to the SSO service at IdP_0. If a security context for the principal already exists at IdP_0, skip to step 10.
4. The `AuthnRequest` is cached and the client is redirected to the terminal IdP (IdP_1).
5. The client makes a SAML `AuthnRequest` to the SSO service at IdP_1. If a security context for the principal does not exist, IdP_1 identifies the principal (details omitted).
6. IdP_1 updates its security context for this principal, issues one or more assertions, and returns a response to the client.
7. The client submits the response to the assertion consumer service at SP_1. The assertion consumer service validates the assertions in the response.
8. SP_1 updates its security context for this principal and redirects the client to IdP_0.
9. The client makes a SAML `AuthnRequest` to IdP_0, the same `AuthnRequest` made at step 3.
10. IdP_0 updates its security context for this principal, issues a single assertion, and returns a response to the client. The response may also contain the assertions issued by IdP_1 at step 6.
11. The client submits the response to the assertion consumer service at SP_0. The assertion consumer service validates the assertions in the response.
12. SP_0 updates its security context for this principal and redirects the client to the resource.
13. The client requests the resource, the same request issued at step 1.
14. The resource makes an access control decision based on the security context for this principal and returns the resource to the client.

There are at least two assertions produced as a result of the previous transaction. Observe that IdP_1 issues an authentication response containing one or more assertions at step 6, which is subsequently consumed by SP_1 at step 7, and likewise IdP_0 issues an authentication response containing a single assertion at step 10, which is consumed by SP_0 at step 11. These authentication responses will contain assertions that themselves contain authentication statements and (optionally) attribute statements. Although IdP_0 is not authoritative for the authentication context and attributes asserted by IdP_1, SP_0 may wish to take all assertions into account to make a fully informed access control decision. We therefore propose a precise packaging of multiple assertion elements below.

Assuming SP_1 exposes the assertions it receives from IdP_1, the IdP Proxy formulates the following assertion at step 10:

```
<!-- SAML1 assertion issued by IdP0 to SP0 -->
<saml:Assertion ...>
  <saml:Conditions ...>...</saml:Conditions>
  <saml:Advice>
         <!-- nested assertion issued by IdP1 to SP1 -->
         <saml:Assertion ...>
                <saml:Conditions ...>...</saml:Conditions>
                <saml:AuthenticationStatement ...>
                  <!-- the subject that authenticated to IdP1 -->
                  <saml:Subject>...</saml:Subject>
                  ...
                </saml:AuthenticationStatement>
                <saml:AttributeStatement>
                  <saml:Subject>...</saml:Subject>
                  ...
                </saml:AttributeStatement>
         </saml:Assertion>
  </saml:Advice>
  <saml:AuthenticationStatement ...>
         <!-- the subject that (indirectly) authenticated to IdP0 -->
         <saml:Subject>...</saml:Subject>
         ...
  </saml:AuthenticationStatement>
  <saml:AttributeStatement>
         <saml:Subject>...</saml:Subject>
         ...
  </saml:AttributeStatement>
</saml:Assertion>
```

More generally, suppose there are *N* IdP Proxies in a chain. Every IdP Proxy in the chain gives rise to an additional level of nesting in the assertion issued to SP_0. In particular, an assertion issued by IdP_j to SP_j has *N* - *j* levels of nesting. SP_j accesses the nested assertion issued by IdP_k ($k > j$) by recursing on the assertion issued by IdP_j precisely *k* - *j* times.