Single Sign-On Implementation Guide

Salesforce, Spring '21

'21

@salesforcedocs

Last updated: April 22, 2021

CONTENTS

SINGLE SIGN-ON

Single sign-on (SSO) is an authentication method that enables users to access multiple applications with one login and one set of credentials. For example, after users log in to your org, they can automatically access all apps from the App Launcher. You can set up your Salesforce org to trust a third-party identity provider to authenticate users. Or you can configure a third-party app to rely on your org for authentication.

When you set up SSO, you configure one system to trust another to authenticate users, eliminating the need for users to log in to each system separately. The system that authenticates users is called an identity provider. The system that trusts the identity provider for authentication is called the service provider.

For example, you can configure Google as an identity provider to authenticate users accessing your org. So users log in to your org using their Google credentials. In this example, your org acts as the service provider, trusting Google to accurately authenticate users.

You can configure your Salesforce org as an identity provider, a service provider, or both. For each of these use cases, you select the authentication protocol to use. Salesforce supports SSO with SAML and OpenID Connect. Salesforce also has preconfigured authentication providers that you can use to enable SSO with systems that have their own authentication protocols, like Facebook. For more information, see Single Sign-On Use Cases. To see a SAML SSO implementation where Salesforce is the identity provider, watch this video.

You can also set up a single identity provider to authenticate users for multiple service providers. For example, you can enable your org as an identity provider and configure Workday and Office 365 as service providers. Users can then access your org, Workday, and Office 365 with one login.

After you configure SSO, set up Single Logout so users can log out of a service provider and identity provider at the same time.

SSO Content

Refer to the following Help articles to learn about and set up SSO in Salesforce.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Federated Authentication is available in: **All** Editions

Delegated Authentication is available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Authentication Providers are available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS

To view the settings:

- View Setup and Configuration

To edit the settings:

- Customize Application

  AND

  Modify All Data

SAML Single Sign-On with Salesforce as the Service Provider

SAML is an open-standard authentication protocol that Salesforce uses for single sign-on (SSO) into a Salesforce org from a third-party identity provider. You can also use SAML to automatically create user accounts with Just-in-Time (JIT) user provisioning.

When you configure Salesforce as the service provider using SAML, authenticated users can flow from a third-party identity provider into Salesforce.

SAML allows your identity provider to exchange user information with Salesforce. When a user tries to log in, your identity provider sends SAML assertions containing facts about the user to Salesforce. Salesforce receives the assertion, validates it against your Salesforce configuration, and allows the user to access your org.

If your users can't log in, review the SAML login history to determine why. Use the SAML Assertion Validator on page 22 to troubleshoot errors in the SAML assertion.

Configuration Help

To configure SSO into your org, establish a SAML identity provider and follow these general steps.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Federated Authentication is available in: **All** Editions

Delegated Authentication is available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Authentication Providers are available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS

To view the settings:

- View Setup and Configuration

To edit the settings:

- Customize Application

  AND

  Modify All Data

CONFIGURE SALESFORCE AS THE SERVICE PROVIDER WITH SAML SINGLE SIGN-ON

Configure Salesforce as a service provider with SAML single sign-on (SSO).

Before you can set up SSO, gather information from your identity provider. Then configure SAML settings for SSO from your corporate identity provider to Salesforce.

1. Set up SSO.
2. Set up an identity provider to encrypt SAML assertions (optional).
3. Enable JIT provisioning (optional). For more information about JIT provisioning, see About Just-In-Time Provisioning.
4. Edit the SAML JIT handler if you selected `Custom SAML JIT with Apex Handler` for JIT provisioning.
5. Test the SSO connection.

Set Up SSO

1. In Salesforce, from Setup, in the Quick Find box, enter *`Single Sign-On Settings`*, then select **Single Sign-On Settings**, and click **Edit**.
2. To view the SAML SSO settings, select `SAML Enabled`.
3. Save your changes.
4. In SAML Single Sign-On Settings, click the appropriate button to create a configuration.
   - **New**–Specify all settings manually.
   - **New from Metadata File**–Import SAML 2.0 settings from an XML file provided by your identity provider. This option uses the XML file to populate as many settings as possible.

     Note: If your XML file contains information for more than one configuration, this option uses the first configuration.

   - **New from Metadata URL**–Import SAML 2.0 settings from a public URL. This option reads the XML file at a public URL and uses it to populate as many settings as possible. To access the URL from your org, add the URL to Remote Site Settings.
5. Give this SSO setting a **Name** for reference.

   Salesforce inserts the corresponding **API Name** value, which you can customize if necessary.
6. Populate the following fields, as appropriate:

Field	Description
`Issuer`	the unique URL that identifies your identity provider in SAML assertions sent to Salesforce.
`Entity ID`	A unique URL that identifies your identity provider as the recipient of SAML requests that Salesforce sends. This entity ID must be the same as the `<saml:Issuer>` attribute in the SAML assertion.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Federated Authentication is available in: **All** Editions

Delegated Authentication is available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Authentication Providers are available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS

To view the settings:
- View Setup and Configuration

To edit the settings:
- Customize Application

  AND

  Modify All Data

Field	Description
`Identity Provider Certificate`	The authentication certificate issued by your identity provider. The certificate size can't exceed 4 KB. If it does, try using a DER encoded file to reduce the size.
`Request Signing Certificate`	The certificate to generate the signature on a SAML request to the identity provider for a service provider-initiated login. This certificate is saved in your **Certificate and Key Management** settings. If you haven't saved a signing certificate, Salesforce uses the global proxy certificate. A signing certificate is preferred because it provides more control over events, such as certificate expiration.
`Request Signature Method`	The hashing algorithm for signed requests, either `RSA-SHA1` or `RSA-SHA256`.
`Assertion Decryption Certificate`	If the identity provider encrypts SAML assertions, select the assertion decryption certificate saved in your **Certificate and Key Management** settings. This field is available only if your org supports multiple SSO configurations. For more information, see Set Up an Identity Provider to Encrypt SAML Assertions.
`SAML Identity Type`	The SAML assertion element that contains the string identifying a Salesforce user. Values include: **`Assertion contains User's Salesforce username`** Use this option if your identity provider passes the Salesforce username in SAML assertions. **`Assertion contains the Federation ID from the User object`** Use this option if your identity provider passes a user identifier for customers or partners in the SAML assertion **`Assertion contains the User ID from the User object`** Use this option if your identity provider passes a user identifier for users from your org in the SAML assertion.
`SAML Identity Location`	The SAML assertion element that specifies where to locate the user's identity. Values include: **`Identity is in the NameIdentifier element of the Subject statement`** The Salesforce `Username` or `FederationIdentifier` is located in the `<Subject>` statement of the assertion. **`Identity is in an Attribute element`** The Salesforce `Username` or `FederationIdentifier` is specified in an `<AttributeValue>`, located in the `<Attribute>` of the assertion.
`Attribute Name`	If you select `Identity is in an Attribute element`, this field contains the value of `AttributeName` that's specified in `<Attribute>`, which is the User ID.
`Name ID Format`	If you select `Identity is in an Attribute element`, this field contains the value for the `nameid-format`. Possible values include `unspecified`, `emailAddress`, or `persistent`. All legal values can be found in the "Name Identifier Format Identifiers" section of the Assertions and Protocols SAML 2.0 specification.
`Service Provider Initiated Request Binding`	If you're using My Domain, choose the binding mechanism your identity provider requests for your SAML messages. Values are: **`HTTP POST`** HTTP POST binding sends SAML messages using base64-encoded HTML forms.

Field	Description
	`HTTP Redirect` HTTP Redirect binding sends base64-encoded and URL-encoded SAML messages within URL parameters. No matter what request binding you select, the SAML response always uses HTTP POST binding.
`Identity Provider Login URL`	The URL where Salesforce sends a SAML request to start the login sequence. If you deployed a My Domain, you can send login requests to the specified URL. If the login URL isn't available, for example, if your identity provider is down, you can return to the standard login page as a backup. To do so, append the <filepath>login</filepath> query string parameter to the URL, for example, `http://mydomain.my.salesforce.com?login`.
`Custom Logout URL`	The URL to direct the user to when they click the **Logout** button in Salesforce. The default is `http://www.salesforce.com`.
`Custom Error URL`	When there's an error during login, specify the URL of the page where users are directed. It must be publicly accessible, such as a public site Visualforce page. The URL can be absolute or relative.

7. Optionally, set up JIT provisioning. For more information, see Enable Just-in-Time user provisioning and About Just-in-Time Provisioning for SAML.
8. Click **Save**.

Click **Download Metadata** to download an XML file of your SAML configuration settings to send to your identity provider. The identity provider can then upload these configuration settings to connect to your org's Experience Cloud site.

Set Up an Identity Provider to Encrypt SAML Assertions

When Salesforce is the service provider for inbound SAML assertions, you can pick a saved certificate to decrypt inbound assertions from third-party identity providers. Provide a copy of this certificate to the identity provider.

1. Open the SSO configuration in the Single Sign-On Settings page.
2. In the `Assertion Decryption Certificate` field, select the certificate for encryption.
3. Set the `SAML Identity Location` to the element where your identifier is located.
4. Provide the Salesforce Login URL value to your identity provider:
   a. Save the SSO configuration. Your SAML settings value for the `Salesforce Login URL` changes.
   b. On the Single Sign-On Settings page, click the name of the SSO configuration.
   c. Copy the new value in the `Salesforce Login URL` field.
5. Provide the identity provider with a copy of the certificate selected in the `Assertion Decryption Certificate` field for encrypting assertions.

Enable JIT Provisioning

1. In Single Sign-On Settings, select `User Provisioning Enabled` in the Just-in-time User Provisioning section.
2. Select a User Provisioning Type:
   - `Standard`—Provisions users automatically using attributes in the SAML assertion.
   - `Custom SAML JIT with Apex handler`—Provisions users based on logic in an Apex class.

     Note: If you're using Professional Edition, you can enable `Standard` JIT provisioning only.
3. If you selected `Standard`, click **Save**, and test the SSO connection. If you selected `Custom SAML JIT with Apex handler`, go to the next step.
4. For `SAML JIT Handler`, select an existing Apex class as the SAML JIT handler class.

   This class must implement the SamlJitHandler interface. If you don't have an Apex class, you can generate one by clicking `Automatically create a SAML JIT handler template`. Edit this class, and modify the default content before you use it. For more information, see Edit the SAML JIT Handler.
5. For `Execute Handler As`, select the user that runs the Apex class. The user must have the Manage Users permission.
6. For `SAML Identity Type`, select `Assertion contains the Federation ID from the User object`. If your identity provider previously used the Salesforce username, tell them to use the Federation ID. JIT provisioning requires a Federation ID as the user type.
7. Click **Save**.

Edit the SAML JIT Handler

Note: If you set up `Standard` JIT provisioning, skip this step and test the SSO connection.

1. From Setup, in the Quick Find box, enter *`Apex Classes`*, then select **Apex Classes**.
2. To map fields between SAML and Salesforce, edit the generated Apex SAML JIT handler to map fields between SAML and Salesforce. In addition, you can modify the generated code to support the following:
   - Custom fields
   - Fuzzy profile matching
   - Fuzzy role matching
   - Contact lookup by email
   - Account lookup by account number
   - Standard user provisioning into an Experience Cloud site
   - Standard user login into an Experience Cloud site
   - Default profile ID usage for portal Just-in-Time provisioning
   - Default portal role usage for portal Just-in-Time provisioning
   - Username generation for portal Just-in-Time provisioning

   For example, to support custom fields in the generated handler code, find the "Handle custom fields here" comment in the generated code. After that code comment, insert your custom field code. For more information and examples, see the SamlJitHandler Interface documentation.

Note: If your identity provider sends JIT attributes for the Contact or Account object in the same assertion as the User object, the generated handler usually can't make updates. For a list of User fields that can't be updated simultaneously with Contact or Account fields, see Objects That Cannot Be Used Together in DML Operations.

Test the SSO Connection

After you configure and save your SAML settings, test them by trying to access the identity provider's application. Your identity provider directs the user's browser to POST a form containing SAML assertions to the Salesforce login page. Each assertion is verified, and if successful, users can log in with SSO.

If you have difficulty using SSO, use the SAML Assertion Validator.

If your users are having problems using SSO, review the SAML login history to determine the problem, and share what you find with your identity provider.

If you're using SAML version 2.0, the OAuth 2.0 Token Endpoint field is populated after you configure SAML. Use the token with the OAuth 2.0 Web Server Flow.

Gather Information from Your Identity Provider

Before you configure SAML settings for single sign-on (SSO) into Salesforce, work with your identity provider to gather SAML information and assertion parameters.

Gather the following information from your identity provider before configuring Salesforce for SAML.

- The unique identifier of the identity provider, known as the issuer ID.
- An authentication certificate.

  Tip: Store the certificate where you can access it from your browser. You upload it to Salesforce when you configure SAML settings.

- The following SAML assertion parameters, as appropriate:
  - The SAML user ID type
  - The SAML user ID location
  - Attribute Name
  - Attribute URI
  - Name ID format

  Note: Attribute Name, Attribute URI, and Name ID format are only necessary if the `SAML User ID Location` is in an Attribute element, and not the name identifier element of a Subject statement.

  Tip: To set up SSO quickly, import SAML 2.0 settings from an XML file (or a URL pointing to the file) on the Single Sign-On Settings page. Obtain the XML file from your identity provider.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Federated Authentication is available in: **All** Editions

Delegated Authentication is available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Authentication Providers are available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS

To view the settings:
- View Setup and Configuration

To edit the settings:
- Customize Application

  AND

  Modify All Data

Customize SAML Start, Login, Logout, and Error Pages

When you configure SAML single sign-on (SSO) into Salesforce, you define URLs for the pages users see throughout the SSO flow. Your identity provider can provide the URLs for the start, login, and logout pages. Or you can provide your own URLs for these pages. You can also specify a custom error page.

You can customize these pages for SAML SSO using external identity providers.

- Identity Provider Login Page—In service provider-initiated SSO, the page where the user is sent for login after trying to access the service provider. The URL for this page must support service provider-initiated SSO and be able to receive SAML requests. Define the URL using the `ssoStartPage` parameter, or enter a URL in `Identity Provider Login URL`. If you specify a login page, we recommend that you also specify a logout page.

  You can use the `RelayState` parameter to control where users are directed after successful login.

- Note: If you use the `RelayState` parameter, maintain its state to send back to the service provider, as recommended in the SAML 2.0 specification. For example, if your identity provider modifies the `RelayState`, such as with URL-decoding, the identity provider must echo back the original `RelayState` to the service provider. Re-encode the relay state before returning it.

- Custom Logout Page—The page the user is redirected to when they click the Logout button or when the user's session expires. To control where the user is redirected, specify a URL for the logout page in `Custom Logout URL`.

  The default logout page is `https://login.salesforce.com`, unless My Domain is enabled. If My Domain is enabled, the default is `https://`*`yourdomain`*`.my.salesforce.com`.

  Or if you configure single logout on page 63, specify a logout page in `Identity Provider Single Logout URL`.

- Custom Error Page (optional)— The page the user is redirected to if a SAML login error occurs. Specify an error page in `Custom Error URL`. The error page must be publicly accessible, like a public site Visualforce page. The URL can be absolute or relative.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Federated Authentication is available in: **All** Editions

Delegated Authentication is available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Authentication Providers are available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS

To view the settings:
- View Setup and Configuration

To edit the settings:
- Customize Application

  AND

  Modify All Data

You can set these values when setting up SSO on page 3. Here's the order of precedence.

1. Session cookie. If you already logged in to Salesforce and a cookie still exists, SSO uses the login and logout pages specified by the session cookie.
2. Values passed in from the identity provider.
3. Values specified on the Single Sign-On Settings Setup page.

If you decide not to specify these values on the Single-Sign-On Settings Setup page, share them with your identity provider. The identity provider uses these values in either the login URL or the SAML assertion.

- `ssoStartPage`–The page that the user is redirected to when trying to log in. This value is the SAML identity provider's login page. The user is directed to this page when requesting a protected resource in Salesforce without an active session.
- `startURL`–The URL to direct the user to when SSO completes successfully. The URL can be absolute, for example, *`https://yourInstance.salesforce.com/001/o`*. Or it can be relative, for example, */001/o*.

You can also use the `RelayState` parameter to control where users are redirected after a successful login. If you use the `RelayState` parameter, make sure that you maintain its state to send back to the service provider, as recommended in the SAML 2.0 specification. For example, if your identity provider modifies the relay state, such as with URL-decoding, the identity provider must echo back the relay state to the service provider. Reencode the relay state before returning it.

- `logoutURL`–The URL where you want the user to be directed when they click the **Logout** button in Salesforce. The default is `http://www.salesforce.com`.

Here's an example of an `<AttributeStatement>` that contains both the `ssoStartPage` and `logoutURL`:

```
<saml:AttributeStatement>
   <saml:Attribute Name="ssoStartPage"
NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
      <saml:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:anyType">
              http://www.customer.org
      </saml:AttributeValue>
   </saml:Attribute>

   <saml:Attribute Name="logoutURL"
NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri">
      <saml:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">
              https://www.salesforce.com
      </saml:AttributeValue>
   </saml:Attribute>
</saml:AttributeStatement>
```

Example SAML Assertions

Salesforce supports several SAML assertion formats sent by your identity provider, with extra requirements for specific features like encrypted assertions and Just-in-Time (JIT) provisioning. To help your identity provider determine the format of SAML assertions to use with your Salesforce org, share these examples.

SAML assertions must be signed according to the XML Signature specification, using RSA and either SHA-1 or SHA-256.

In addition to the general single sign-on (SSO) examples, use the following samples for the specific feature:

- assertions for encrypted SAML
- assertions for portals
- assertions for Sites
- assertion for JIT provisioning

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Federated Authentication is available in: **All** Editions

Delegated Authentication is available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Authentication Providers are available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS

To view the settings:
- View Setup and Configuration

To edit the settings:
- Customize Application

  AND

  Modify All Data

General SSO Examples

SAML User ID type is the Salesforce username, and SAML User ID location is the `<NameIdentifier>` element in the `<Subject>` element

```
<saml:Subject>
    <saml:NameID
Format="urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified">user101@salesforce.com</saml:NameID>

    <saml:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
        <saml:SubjectConfirmationData NotOnOrAfter="2008-06-26T02:44:24.173Z"
Recipient="http://localhost:9000"/>
    </saml:SubjectConfirmation>
</saml:Subject>
```

SAML User ID type is the Salesforce username, and SAML User ID location is the `<Attribute>` element

```
<saml:AttributeStatement>
    <saml:Attribute FriendlyName="fooAttrib" Name="SFDC_USERNAME"
NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
        <saml:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
```

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">
            user101@salesforce.com
        </saml:AttributeValue>
    </saml:Attribute>
 </saml:AttributeStatement>
```

SAML User ID type is the Salesforce User object's `FederationIdentifier` field, and SAML User ID location is the `<NameIdentifier>` element in the `<Subject>` element

```
<saml:Subject>
    <saml:NameID
Format="urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified">MyName</saml:NameID>
    <saml:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
        <saml:SubjectConfirmationData NotOnOrAfter="2008-06-26T02:48:25.730Z"
Recipient="http://localhost:9000/"/>
    </saml:SubjectConfirmation>
</saml:Subject>
```

Note: The name identifier can be any arbitrary string, including email addresses or numeric ID strings.

SAML User ID type is theSalesforce User object's `FederationIdentifier` field, and SAML User ID location is the `<Attribute>` element

```
<saml:AttributeStatement>
    <saml:Attribute FriendlyName="fooAttrib" Name="SFDC_ATTR"
NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
        <saml:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">
            user101
        </saml:AttributeValue>
    </saml:Attribute>
</saml:AttributeStatement>
```

SAML User ID type is the Salesforce username, and SAML User ID location is the `<NameIdentifier>` element in the `<Subject>` element

Here's a complete SAML response:

```
<samlp:Response ID="_257f9d9e9fa14962c0803903a6ccad931245264310738"
   IssueInstant="2009-06-17T18:45:10.738Z" Version="2.0">
<saml:Issuer Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">
   https://www.salesforce.com
</saml:Issuer>

<samlp:Status>
   <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
</samlp:Status>

<saml:Assertion ID="_3c39bc0fe7b13769cab2f6f45eba801b1245264310738"
   IssueInstant="2009-06-17T18:45:10.738Z" Version="2.0">
   <saml:Issuer Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">
      https://www.salesforce.com
   </saml:Issuer>

   <saml:Signature>
      <saml:SignedInfo>
```

```
         <saml:CanonicalizationMethod
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
        <saml:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>

         <saml:Reference URI="#_3c39bc0fe7b13769cab2f6f45eba801b1245264310738">
            <saml:Transforms>
               <saml:Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
               <saml:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
                  <ec:InclusiveNamespaces PrefixList="ds saml xs"/>
               </saml:Transform>
            </saml:Transforms>
            <saml:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
            <saml:DigestValue>vzR9Hfp8d16576tEDeq/zhpmLoo=
            </saml:DigestValue>
         </saml:Reference>
      </saml:SignedInfo>
      <saml:SignatureValue>
         AzID5hhJeJlG2llUDvZswNUrlrPtR7S37QYH2W+Un1n8c6kTC
         Xr/lihEKPcA2PZt86eBntFBVDWTRlh/W3yUgGOqQBJMFOVbhK
         M/CbLHbBUVT5TcxIqvsNvIFdjIGNkf1W0SBqRKZOJ6tzxCcLo
         9dXqAyAUkqDpX5+AyltwrdCPNmncUM4dtRPjI05CL1rRaGeyX
         3kkqOL8p0vjm0fazU5tCAJLbYuYgU1LivPSahWNcpvRSlCI4e
         Pn2oiVDyrcc4et12inPMTc2lGIWWWWJyHOPSiXRSkEAIwQVjf
         Qm5cpli44Pv8FCrdGWpEE0yXsPBvDkM9jIzwCYGG2fKaLBag==
      </saml:SignatureValue>
      <saml:KeyInfo>
         <saml:X509Data>
            <saml:X509Certificate>
               MIIEATCCAumgAwIBAgIBBTANBgkqhkiG9w0BAQ0FADCBgzELM
               [Certificate truncated for readability...]
            </saml:X509Certificate>
         </saml:X509Data>
      </saml:KeyInfo>
   </saml:Signature>

   <saml:Subject>
      <saml:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified">
         saml01@salesforce.com
      </saml:NameID>

      <saml:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
      <saml:SubjectConfirmationData NotOnOrAfter="2009-06-17T18:50:10.738Z"
         Recipient="https://login.salesforce.com"/>
      </saml:SubjectConfirmation>
   </saml:Subject>

   <saml:Conditions NotBefore="2009-06-17T18:45:10.738Z"
      NotOnOrAfter="2009-06-17T18:50:10.738Z">

      <saml:AudienceRestriction>
         <saml:Audience>https://saml.salesforce.com</saml:Audience>
      </saml:AudienceRestriction>
   </saml:Conditions>
```

```
   <saml:AuthnStatement AuthnInstant="2009-06-17T18:45:10.738Z">
      <saml:AuthnContext>
         <saml:AuthnContextClassRef>urn:oasis:names:tc:SAML:2.0:ac:classes:unspecified
         </saml:AuthnContextClassRef>
      </saml:AuthnContext>
   </saml:AuthnStatement>

   <saml:AttributeStatement>

      <saml:Attribute Name="portal_id">
         <saml:AttributeValue xsi:type="xs:anyType">060D00000000SHZ
         </saml:AttributeValue>
      </saml:Attribute>

      <saml:Attribute Name="organization_id">
         <saml:AttributeValue xsi:type="xs:anyType">00DD0000000F7L5
         </saml:AttributeValue>
      </saml:Attribute>

      <saml:Attribute Name="ssostartpage"
         NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">

         <saml:AttributeValue xsi:type="xs:anyType">
            http://www.salesforce.com/security/saml/saml20-gen.jsp
         </saml:AttributeValue>
      </saml:Attribute>

      <saml:Attribute Name="logouturl"
         NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri">

         <saml:AttributeValue xsi:type="xs:string">
            http://www.salesforce.com/security/del_auth/SsoLogoutPage.html
         </saml:AttributeValue>
      </saml:Attribute>
   </saml:AttributeStatement>
</saml:Assertion>
</samlp:Response>
```

Example Assertions for Encrypted SAML

These examples are useful if you set up your org to decrypt encrypted SAML assertions from your identity provider.

The expected tag for an encrypted assertion is `<EncryptedAssertion>`.

If you set up encrypted assertions, your identity provider must encrypt the entire assertion. However, Salesforce only supports one layer of encryption. For example, you can't encrypt `<NameID>` to `<EncryptedID>`, and then encrypt the whole assertion.

Here's an example of an encrypted SAML assertion with `<EncryptedKey>` outside of `<EncryptedData>`.

```
<saml:EncryptedAssertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
   <xenc:EncryptedData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#" Id="Encrypted_DATA_ID"
    Type="http://www.w3.org/2001/04/xmlenc#Element">
      <xenc:EncryptionMethod   Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
      <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
```

```
         <ds:RetrievalMethod URI="#Encrypted_KEY_ID"
Type="http://www.w3.org/2001/04/xmlenc#EncryptedKey"/>
      </ds:KeyInfo>
      <xenc:CipherData >
         <xenc:CipherValue>Nk4W4mx...</xenc:CipherValue>
      </xenc:CipherData>
   </xenc:EncryptedData>
   <xenc:EncryptedKey xmlns:xenc="http://www.w3.org/2001/04/xmlenc#" Id="Encrypted_KEY_ID">

      <xenc:EncryptionMethod   Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
      <xenc:CipherData>
         <xenc:CipherValue>PzA5X...</xenc:CipherValue>
      </xenc:CipherData>
      <xenc:ReferenceList>
         <xenc:DataReference URI="#Encrypted_DATA_ID"/>
      </xenc:ReferenceList>
   </xenc:EncryptedKey>
</saml:EncryptedAssertion>
```

Here's an example of an encrypted SAML assertion with `<EncryptedKey>` inside `<EncryptedData>`.

```
<xenc:EncryptedData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
Type="http://www.w3.org/2001/04/xmlenc#Element">
   <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#aes256-cbc" />
  <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
   <xenc:EncryptedKey xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
    <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5" />
    <xenc:CipherData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
     <xenc:CipherValue>HWm..</xenc:CipherValue>
    </xenc:CipherData>
   </xenc:EncryptedKey>
   …
 </xenc:EncryptedData>
```

Example SAML Assertions for Portals

The following is a portion of a SAML assertion statement showing the `portal_id` and `organization_id` attributes:

```
<saml:AttributeStatement>
   <saml:Attribute Name="portal_id">
      <saml:AttributeValue xsi:type="xs:anyType">060D00000000SHZ</saml:AttributeValue>
   </saml:Attribute>

   <saml:Attribute Name="organization_id">
         <saml:AttributeValue xsi:type="xs:anyType">00DD0000000F7P5</saml:AttributeValue>

   </saml:Attribute>
</saml:AttributeStatement>
```

Here's a complete SAML assertion statement for SSO for portals. The example org uses federated sign-on, which is included in an attribute (see the `<saml:AttributeStatement>` in bold text in the assertion), not in the subject.

```
<samlp:Response ID="_f97faa927f54ab2c1fef230eee27cba21245264205456"
      IssueInstant="2009-06-17T18:43:25.456Z" Version="2.0">
```

```
   <saml:Issuer Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">
      https://www.salesforce.com</saml:Issuer>

   <samlp:Status>
      <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success"/>
   </samlp:Status>

   <saml:Assertion ID="_f690da2480a8df7fcc1cbee5dc67dbbb1245264205456"
      IssueInstant="2009-06-17T18:45:10.738Z" Version="2.0">
      <saml:Issuer Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">
         https://www.salesforce.com
      </saml:Issuer>

      <saml:Signature>
         <saml:SignedInfo>
            <saml:CanonicalizationMethod
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
           <saml:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>

            <saml:Reference URI="#_f690da2480a8df7fcc1cbee5dc67dbbb1245264205456">
               <saml:Transforms>
                  <saml:Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
                  <saml:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
                     <ec:InclusiveNamespaces PrefixList="ds saml xs"/>
                  </saml:Transform>
               </saml:Transforms>
               <saml:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
               <saml:DigestValue>vzR9Hfp8d16576tEDeq/zhpmLoo=
               </saml:DigestValue>
            </saml:Reference>
         </saml:SignedInfo>
         <saml:SignatureValue>
            AzID5hhJeJlG2llUDvZswNUrlrPtR7S37QYH2W+Un1n8c6kTC
            Xr/lihEKPcA2PZt86eBntFBVDWTRlh/W3yUgGOqQBJMFOVbhK
            M/CbLHbBUVT5TcxIqvsNvIFdjIGNkf1W0SBqRKZOJ6tzxCcLo
            9dXqAyAUkqDpX5+AyltwrdCPNmncUM4dtRPjI05CL1rRaGeyX
            3kkqOL8p0vjm0fazU5tCAJLbYuYgU1LivPSahWNcpvRSlCI4e
            Pn2oiVDyrcc4et12inPMTc2lGIWWWWJyHOPSiXRSkEAIwQVjf
            Qm5cpli44Pv8FCrdGWpEE0yXsPBvDkM9jIzwCYGG2fKaLBag==
         </saml:SignatureValue>
         <saml:KeyInfo>
            <saml:X509Data>
               <saml:X509Certificate>
                  MIIEATCCAumgAwIBAgIBBTANBgkqhkiG9w0BAQ0FADCBgzELM
                  Certificate truncated for readability...
               </saml:X509Certificate>
            </saml:X509Data>
         </saml:KeyInfo>
      </saml:Signature>

      <saml:Subject>
         <saml:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified">null
```

```
   </saml:NameID>

   <saml:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
   <saml:SubjectConfirmationData NotOnOrAfter="2009-06-17T18:48:25.456Z"
      Recipient="https://login.salesforce.com/?saml=02HKiPoin4f49GRMsOdFmhTgi
      _0nR7BBAflopdnD3gtixujECWpxr9klAw"/>
      </saml:SubjectConfirmation>
</saml:Subject>

<saml:Conditions NotBefore="2009-06-17T18:43:25.456Z"
   NotOnOrAfter="2009-06-17T18:48:25.456Z">

   <saml:AudienceRestriction>
      <saml:Audience>https://saml.salesforce.com</saml:Audience>
   </saml:AudienceRestriction>
</saml:Conditions>

<saml:AuthnStatement AuthnInstant="2009-06-17T18:43:25.456Z">

   <saml:AuthnContext>
      <saml:AuthnContextClassRef>urn:oasis:names:tc:SAML:2.0:ac:classes:unspecified

      </saml:AuthnContextClassRef>
   </saml:AuthnContext>
</saml:AuthnStatement>

<saml:AttributeStatement>

   <saml:Attribute FriendlyName="Friendly Name" Name="federationId"
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
      <saml:AttributeValue xsi:type="xs:string">saml_portal_user_federation_id
      </saml:AttributeValue>
      <saml:AttributeValue xsi:type="xs:string">SomeOtherValue
      </saml:AttributeValue>
   </saml:Attribute>

   <saml:Attribute Name="portal_id">
      <saml:AttributeValue xsi:type="xs:anyType">060D00000000SHZ
      </saml:AttributeValue>
   </saml:Attribute>

   <saml:Attribute Name="organization_id">
      <saml:AttributeValue xsi:type="xs:anyType">00DD0000000F7Z5
      </saml:AttributeValue>
   </saml:Attribute>

   <saml:Attribute Name="ssostartpage"
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">

      <saml:AttributeValue xsi:type="xs:anyType">
         http://www.salesforce.com/qa/security/saml/saml20-gen.jsp
      </saml:AttributeValue>
   </saml:Attribute>
```

```
         <saml:Attribute Name="logouturl"
            NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri">

            <saml:AttributeValue xsi:type="xs:string">
                http://www.salesforce.com/qa/security/del_auth/SsoLogoutPage.html
            </saml:AttributeValue>
         </saml:Attribute>
      </saml:AttributeStatement>
   </saml:Assertion>
</samlp:Response>
```

Example SAML Assertion for Sites

The following SAML assertion statement shows the `portal_id`, `organization_id`, and `siteurl` attributes:

```
<saml:AttributeStatement>
   <saml:Attribute Name="portal_id">
      <saml:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:type="xs:anyType">060900000004cDk
      </saml:AttributeValue>
   </saml:Attribute>
   <saml:Attribute Name="organization_id">
      <saml:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:type="xs:anyType">00D900000008bX0
      </saml:AttributeValue></saml:Attribute>
   <saml:Attribute Name="siteurl">
      <saml:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:type="xs:anyType">https://ap1.force.com/mySuffix</saml:AttributeValue>
   </saml:Attribute>
</saml:AttributeStatement>
```

Example SAML Assertion for JIT Provisioning

Here's an example SAML assertion for JIT provisioning.

```
<saml:AttributeStatement>

   <saml:Attribute Name="User.Username"
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
      <saml:AttributeValue xsi:type="xs:anyType">testuser@123.org
      </saml:AttributeValue>
   </saml:Attribute>

   <saml:Attribute Name="User.Phone"
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
      <saml:AttributeValue xsi:type="xs:anyType">415-123-1234
      </saml:AttributeValue>
   </saml:Attribute>

   <saml:Attribute Name="User.FirstName"
```

```
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
   <saml:AttributeValue xsi:type="xs:anyType">Testuser
   </saml:AttributeValue>
</saml:Attribute>

<saml:Attribute Name="User.LanguageLocaleKey"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
   <saml:AttributeValue xsi:type="xs:anyType">en_US
   </saml:AttributeValue>
</saml:Attribute>

<saml:Attribute Name="User.CompanyName"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
   <saml:AttributeValue xsi:type="xs:anyType">Salesforce.com
   </saml:AttributeValue>
</saml:Attribute>

<saml:Attribute Name="User.Alias"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
   <saml:AttributeValue xsi:type="xs:anyType">tlee2
   </saml:AttributeValue>
</saml:Attribute>

<saml:Attribute Name="User.CommunityNickname"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
   <saml:AttributeValue xsi:type="xs:anyType">tlee2
   </saml:AttributeValue>
</saml:Attribute>

<saml:Attribute Name="User.UserRoleId"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
   <saml:AttributeValue xsi:type="xs:anyType">000000000000000
   </saml:AttributeValue>
</saml:Attribute>

<saml:Attribute Name="User.Title"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
   <saml:AttributeValue xsi:type="xs:anyType">Mr.
   </saml:AttributeValue>
</saml:Attribute>

<saml:Attribute Name="User.LocaleSidKey"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
   <saml:AttributeValue xsi:type="xs:anyType">en_CA
   </saml:AttributeValue>
</saml:Attribute>

<saml:Attribute Name="User.Email"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
   <saml:AttributeValue xsi:type="xs:anyType">testuser@salesforce.com
   </saml:AttributeValue>
</saml:Attribute>

<saml:Attribute Name=" User.FederationIdentifier"
```

```
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
      <saml:AttributeValue xsi:type="xs:anyType">tlee2
      </saml:AttributeValue>
   </saml:Attribute>

   <saml:Attribute Name="User.TimeZoneSidKey"
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
      <saml:AttributeValue xsi:type="xs:anyType">America/Los_Angeles
      </saml:AttributeValue>
   </saml:Attribute>

   <saml:Attribute Name="User.LastName"
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
      <saml:AttributeValue xsi:type="xs:anyType">Lee
      </saml:AttributeValue>
   </saml:Attribute>

   <saml:Attribute Name="User.ProfileId"
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
      <saml:AttributeValue xsi:type="xs:anyType">00ex0000001pBNL
      </saml:AttributeValue>
   </saml:Attribute>

   <saml:Attribute Name="User.IsActive"
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
      <saml:AttributeValue xsi:type="xs:anyType">1
      </saml:AttributeValue>
   </saml:Attribute>

   <saml:Attribute Name="User.EmailEncodingKey"
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
      <saml:AttributeValue xsi:type="xs:anyType">UTF-8
      </saml:AttributeValue>
   </saml:Attribute>

</saml:AttributeStatement>
```

VIEW AND EDIT SINGLE SIGN-ON SETTINGS

After you've configured your Salesforce org to use SAML, you can manage the SAML configuration from the Single Sign-On Settings page.

From Setup, enter *Single Sign-On Settings* in the Quick Find box, then select **Single Sign-On Settings**.

After the SAML configuration completes, the Single Sign-On Settings page displays the generated URLs and OAuth 2.0 token endpoint.

Field	Description
Salesforce Login URL	For SAML 2.0. The URL associated with the login for the Web SSO OAuth assertion flow. This URL appears if you configured SAML with "Assertion contains the User's Salesforce username" for SAML Identity Type and "Identity is in the NameIdentifier element of the Subject statement" for SAML Identity Location.
Salesforce Logout URL	For SAML 2.0. The Salesforce logout URL that users are directed to after they log off. This URL appears if you didn't specify a value for Custom Logout URL.
OAuth 2.0 Token Endpoint	For SAML 2.0. The ACS URL used when enabling Salesforce as an identity provider in the Web SSO OAuth assertion flow.

From this page you can do any of the following:

- Click **Edit** to change the existing SAML configuration.
- Click **SAML Assertion Validator** to validate the SAML settings for your org using a SAML assertion provided by your identity provider.
- Click **Download Metadata** to download an XML file of your SAML configuration settings to send to your identity provider. The identity provider can then upload these configuration settings to connect to your org's Experience Cloud site. Enabled only if your identity provider supports metadata and if you are using SAML 2.0.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Federated Authentication is available in: **All** Editions

Delegated Authentication is available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Authentication Providers are available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS

To view the settings:

- View Setup and Configuration

To edit the settings:

- Customize Application

  AND

  Modify All Data

TROUBLESHOOT SAML ASSERTION ERRORS

Use the SAML Assertion Validator to troubleshoot single sign-on (SSO) login problems and identify errors in SAML assertions sent by your identity provider.

If users have difficulty logging in after you configure Salesforce as a SAML service provider, use the SAML Assertion Validator to find assertion errors.

1. Obtain a SAML assertion in plain XML, base-64 encoded, or deflated and base-64 encoded format from your identity provider.

   If a user can't log in to Salesforce, the invalid SAML assertion is automatically entered into the SAML Assertion Validator, if possible. Some errors prevent the assertion from being entered automatically.

2. From Setup, enter *Single Sign-On Settings* in the Quick Find box, select **Single Sign-On Settings**, then click **SAML Assertion Validator**.
3. Enter the SAML assertion into the text box, and click **Validate**.
4. Note: If your org has multiple SAML SSO configurations, the validator tries to detect the right one. You can also select a configuration by clicking the dropdown arrow next to **Auto detect config**.
5. Share the results of the validation errors with your identity provider.

Note: Some errors stop the validator from continuing, potentially leaving undetected errors. After you fix initial errors, run the assertion through the validator again to ensure that it hasn't missed anything.

The validator only detects errors related to the SAML assertion. To troubleshoot errors unrelated to the assertion, view the login history.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Federated Authentication is available in: **All** Editions

Delegated Authentication is available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Authentication Providers are available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS

To view the settings:

- View Setup and Configuration

To edit the settings:

- Customize Application

  AND

  Modify All Data

SAML Login Errors

If users have trouble accessing your org with single sign-on (SSO), use the login history to determine whether it's a SAML assertion error or a configuration problem. If it's an assertion-related error, identify specific assertion problems with the SAML Assertion Validator. Work with your identity provider to ensure that both the SAML assertion and your SSO configuration are valid.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Federated Authentication is available in: **All** Editions

Delegated Authentication is available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Authentication Providers are available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS

To view the settings:
- View Setup and Configuration

To edit the settings:
- Customize Application

  AND

  Modify All Data

Login History

Use the login history on page 26 to determine whether a login error is related to the SAML assertion or to your SSO configuration.

If you see any of the following errors in the login history, use the SAML Assertion Validator on page 22 to find the specific error in the assertion:

Assertion Expired
The timestamp on the assertion is too old.

Assertion Invalid
Something is wrong with the assertion, like a missing `<Subject>` element.

Audience Invalid
The value specified in `<Audience>` doesn't match the `Entity ID` you specified during SSO configuration.

If you see any of the following errors in the login history, check your SSO settings for a configuration problem. From Setup, in the Quick Find box, enter *`Single Sign-On Settings`*, and then select **Single Sign-On Settings**. Get a sample SAML assertion from your identity provider, and confirm that you have the right information in your configuration. If your configuration is correct, run the sample assertion through the SAML Assertion Validator.

Configuration Error/Perm Disabled
Something is wrong with your SAML configuration in Salesforce. For example, the certificate that you uploaded is corrupt, or you disabled SAML in your org's Single Sign-On Settings.

Issuer Mismatched
Check that the issuer specified in your configuration matches the issuer in the assertion.

Recipient Mismatched
Check that the recipient specified in your configuration matches the recipient in the assertion.

Replay Detected
Every assertion has a unique ID. This error means that Salesforce detected a repeat assertion ID.

Signature Invalid
The certificate that you uploaded during configuration failed to validate the signature in the assertion. Work with your identity provider to confirm that you have the right certificate.

Subject Confirmation Error
Check that the `<Subject>` specified in your configuration matches the `<Subject>` in the assertion.

SAML Assertion Validator

When you run the SAML Assertion Validator on page 22, it checks the assertion against Salesforce's validity requirements and tells you whether the assertion met each requirement. Salesforce imposes the following validity requirements on assertions, shown here in the order they appear on the results page:

Status
The status field in the SAML response must indicate success.

Authentication Statement
The identity provider must include an `<AuthenticationStatement>` in the assertion.

Conditions Statement
Some assertions use a `<Conditions>` statement to constrain the time period when the assertion is valid, called the validity period. If the assertion contains a `<Conditions>` statement with a timestamp, the timestamp must be valid.

Timestamps
The identity provider generates a timestamp to indicate when it sent the assertion. Salesforce must receive the assertion from your identity provider within 5 minutes of the timestamp, plus or minus 3 minutes. In practice, this constraint means Salesforce can receive the assertion up to 8 minutes after the timestamp or 3 minutes before it. If the assertion specifies a shorter validity period, the validator checks this requirement too. The `NotBefore` and `NotOnOrAfter` constraints must also be defined and valid.

Attribute
If you set your Salesforce configuration to `Identity is in an Attribute element`, the assertion from the identity provider must contain an `<AttributeStatement>`.

Only `<AttributeName>` is required.

Format
The `Format` attribute of an `<Issuer>` statement must be set to `"urn:oasis:names:tc:SAML:2.0:nameid-format:entity"` or not set at all.

For example:

```
<saml:Issuer
Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">https://www.salesforce.com</saml:Issuer>
```

This example is also valid:

```
<saml:Issuer >https://www.salesforce.com</saml:Issuer>
```

Issuer
The issuer specified in the assertion must match the issuer you specified when you configured Salesforce.

Subject
The subject of the assertion must be either the Salesforce username or the Federation ID of the user.

Audience
The assertion must contain an `<Audience>` value that matches the `Entity ID` you specified during configuration. The default value is `https://saml.salesforce.com`.

Recipient
The assertion must contain a recipient that matches either the Salesforce login URL you specified in the Salesforce configuration or the OAuth 2.0 token endpoint.

Signature
The assertion must include a valid signature. The signature must be created using the private key associated with the certificate you uploaded when you configured SSO. We recommend that you sign the assertion and the response.

Site URL Attribute

Valid values are:

- Not Provided
- Checked
- Site URL is invalid
- HTTPS is required for Site URL
- The specified Site is inactive or has exceeded its page limit

Portal and Organization ID

Optional. For SAML login into a portal, the recipient and organization ID in the assertion must match the recipient and organization ID specified in your SSO configuration.

Session Security Level

Optional. Session security level describes how secure a user session is. If you configured session security settings, the session security level in the assertion must match your configuration.

REVIEW THE LOGIN HISTORY

When users fail to log in to your org with single sign-on (SSO), search the login history to find out why. For example, see if a login failure is related to the SAML assertion or to your Salesforce configuration.

When a user logs in to your org from an external SAML identity provider, like Okta, the identity provider sends SAML assertions with user information to Salesforce. Salesforce validates the SAML assertion and its user claims with information from your SSO configuration and the users in your org. If the SAML assertion is invalid or something is wrong with your SAML configuration, the user fails to log in. To see why the login failed, view the login history.

Use the login history to determine whether an error is related the SAML assertion or to your Salesforce configuration. If the error is related to the SAML assertion, use the SAML Assertion Validator to locate specific errors in the assertion.

For example, you configure SSO into your org from Okta, your identity provider, and ask a few users to test it by logging in. All test users fail to log in. You go to Setup, review the login history, and find the failed attempts. Under `Status`, you see the same error message for all test users: `Assertion Invalid`. It's a SAML assertion problem, so you go to the SAML Assertion Validator, which contains the assertion from the most recent failed attempt. You run the validator, locate the error, and work with Okta to fix it.

Or, consider the same scenario with a different error message indicating a problem with your Salesforce configuration: `Issuer Mismatched`. This error message tells you that the issuer specified in Okta's assertion is different than the issuer you specified in your Salesforce configuration. You obtain the correct issuer from Okta, edit your SSO settings on page 21, and fix the error.

For a list of reasons for login failures, see SAML Login Errors on page 23.

To view the login history:

1. From Setup, enter *`Login History`* in the Quick Find box, then select **Login History**. You can view and download your org's login history for the last 6 months.
2. To customize the information the login history shows, from the Login History page, click **Create New View**. For more information on creating login history views, see Monitor Login History.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Federated Authentication is available in: **All** Editions

Delegated Authentication is available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Authentication Providers are available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS

To view the settings:

- View Setup and Configuration

To edit the settings:

- Customize Application

  AND

  Modify All Data

JUST-IN-TIME PROVISIONING FOR SAML

Use Just-in-Time (JIT) provisioning to automatically create a user account in your Salesforce org the first time a user logs in with single sign-on (SSO). JIT provisioning can reduce your workload and save time. JIT provisioning also automatically applies password policies for your corporate network to your org, potentially increasing security.

With JIT provisioning, you can use a SAML assertion to create users the first time they log in to your org from a third-party identity provider. JIT provisioning saves you time and effort because it eliminates the need to provision users or create user accounts in advance.

For example, your company adds several new employees, and you want to create user accounts for them in your org as soon as possible. You configure SSO and set up JIT provisioning. Now, when the new employees log in with SSO, the JIT provisioning method automatically creates their accounts.

JIT provisioning works with your identity provider to pass user information to Salesforce in a SAML 2.0 assertion. You can create and modify accounts this way. Configure SAML settings for SSO on page 3 in your org before you set up JIT provisioning.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **All** Editions

Just-in-Time Provisioning Requirements and SAML Assertion Fields

With Just-in-Time (JIT) provisioning, the identity provider passes user information to your Salesforce org in a SAML assertion to automatically create user accounts. Work with your identity provider to determine which user information you want to pass to your org.

Your identity provider sends user information to your org in an `Attributes` statement in the SAML assertion. Work with your identity provider to ensure the `Attributes` statement is formatted correctly. For a sample assertion, see Sample SAML Assertion for Just-in-Time Provisioning on page 18.

Note: `Provision Version` is supported as an optional attribute. If it isn't specified, the default is 1.0. For example:

```
<saml:Attribute Name="ProvisionVersion" NameFormat=
   "urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
      <saml:AttributeValue xsi:type="xs:anyType">1.0</saml:AttributeValue>
</saml:Attribute>
```

SAML Assertion Fields

When a user logs in to an org with standard JIT provisioning enabled, Salesforce pulls user data from the identity provider and stores it in a new `User` object. To tell Salesforce to create this object, you must use the `User.` prefix for all `User` object fields in the SAML assertion. In this example, the `User.` prefix is added to the `Username` field name.

```
<saml:Attribute
   Name="User.Username"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
      <saml:AttributeValue xsi:type="xs:anyType">testuser@123.org</saml:AttributeValue>
</saml:Attribute>
```

For detailed information about each `User` object field, see *SOAP API Developer Guide*: User.

These User fields are required:

Email

LastName

ProfileId

A profile ID is different in each org, even for a standard profile. To make it easier to find the profile name, pass the `ProfileName` into the `ProfileId` field.

Username (insert only)

With JIT, you can pass a new username into the `User.Username` field. You can also specify the `User.FederationIdentifier` if it is present. However, the `Username` and `FederationIdentifier` fields can't be updated through the SOAP API.

These supported User fields are optional:

FirstName

CommunityNickname

FederationIdentifier

TimeZoneSidKey

LanguageLocaleKey

LocaleSidKey

EmailEncodingKey

DefaultCurrencyIsoCode

Role

Alias

Title

Phone

CompanyName

Active

AboutMe

Street

State

City

Zip

Country

ReceivesAdminInfoEmails

ForecastEnabled

CallCenter

Manager

MobilePhone

DelegatedApproverId

Department

Division

EmployeeNumber

Extension

Fax

ReceivesInfoEmails

If you include custom fields in the SAML assertion:

- Only text type custom fields are supported.
- Only the `insert` and `update` functions are supported for custom fields.

Just-in-Time Provisioning and SAML Assertion Fields for Portals

With Just-in-Time (JIT) provisioning for portals, you can use a SAML assertion to create customer and partner portal users the first time they log in from an identity provider. Customer portals and partner portals are not available for new orgs as of the Summer '13 release. Use Experience Cloud instead.

Creating Portal Users

The `Portal ID` and `Organization ID` must be specified as part of the SAML assertion. You can find both of these parameters on the company information page for the org or portal. Because you can also provision regular users, the `Portal ID` is used to distinguish between a regular and portal JIT provisioning request. If no `Portal ID` is specified, then the request is treated as a JIT request for regular platform user. Here are the requirements for a creating a portal user.

- Specify a `Federation ID`. If the ID belongs to an existing user account, the user account is updated. If the user account is inactive, the user account is updated but left inactive unless `User.IsActive` in the JIT assertion is set to true. If there is no user account with that `Federation ID`, the system creates a user.
- If the portal isn't self-registration enabled and a default new user profile and role aren't specified, the `User.ProfileId` field must contain a valid profile name or ID. In addition, the `User.PortalRole` field must contain a valid portal role name or ID. Use `Worker` for all portal users.

  Note: The `User.Role` must be null.

Creating and Modifying Accounts

Create or modify an account by specifying a valid `Account ID` or both the `Account.AccountNumber` and `Account.Name`.

- Matching is based on `Account.AccountNumber`. If multiple accounts are found, an error is displayed. Otherwise, the account is updated.
- If no matching account is found, one is created.
- You must specify the `Account.Owner` in the SAML assertion and ensure that the field level security for the `Account.AccountNumber` field is set to visible for this owner's profile.

Creating and Modifying Contacts

Create or modify a contact by specifying a valid contact ID in `User.Contact` or both the `Contact.Email` and `Contact.LastName`.

- Matching is based on `Contact.Email`. If multiple contacts are found, an error is displayed. Otherwise, the contact is updated.
- If no matching contact is found, one is created.

Supported Fields for the Portal SAML Assertion

To correctly identify which object to create in Salesforce, you must use a prefix. In the SAML assertion, use the `Account` prefix for all fields in the Account schema, for example, `Account.AccountId`. And use the `Contact` prefix for all fields in the Contact schema. In this example, the `Contact` prefix has been added to the `Email` field name.

```
<saml:Attribute
   Name="Contact.Email"
```

```
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
      <saml:AttributeValue xsi:type="xs:anyType">testuser@123.org</saml:AttributeValue>
</saml:Attribute>
```

In addition to the standard User attributes supported for regular SAML JIT users, these Account attributes are also supported. For example, specifying an `Account.Phone` attribute in the assertion updates the account's Phone field on the corresponding Account object.

Name

AccountNumber

BillingCity

BillingCountry

BillingPostalCode

BillingState

BillingStreet

Owner
The Owner field on the Account object is Account.OwnerId in the API.

AnnualRevenue

Description

NumberOfEmployees

Fax

Industry

Ownership

Phone

Rating

ShippingAddress
The Shipping Address field is a compound field.

ShippingCity

ShippingCountry

ShippingPostalCode

ShippingState

ShippingStreet

Sic

TickerSymbol

Website

These Contact attributes are supported.

Account
This value is the Account Name field on the Contact object and Account.Name in the API.

Email

FirstName

LastName

Phone

CanAllowPortalSelfReg

AssistantName

AssistantPhone

Birthdate

Owner
This value is the Contact Owner field on the Contact object and Contact.OwnerId in the API.

Department

Description

DoNotCall

HasOptedOutOfEmail

Fax

HasOptedOutOfFax

HomePhone

LastCUUpdatetDate
This value is the Last Modified By field on the Contact object and Contact.LastModifiedDate in the API.

LeadSource

MailingAddress
The Mailing Address field is a compound field.

MailingCity

MailingCountry

MailingPostalCode

MailingState

MailingStreet

MobilePhone

Salutation

OtherAddress
The Other Address field is a compound field.

OtherCity

OtherCountry

OtherPostalCode

OtherState

OtherStreet

OtherPhone

Title

These additional User attributes are supported for portal users.

AccountId

ContactId

PortalRole

Use `Worker` for all portal users.

Just-in-Time Provisioning for Experience Cloud

With Just-in-Time (JIT) provisioning for Experience Cloud, you can use a SAML assertion to create Experience Cloud site users the first time they log in from an identity provider.

SAML Single Sign-On Settings

Follow the instructions for Configure Salesforce as the Service Provider with SAML Single Sign-On. Set the values for your configuration, as needed, and include the following values specific to your site for JIT provisioning.

1. Check `User Provisioning Enabled`.
   - Just-in-time provisioning requires a Federation ID in the user type. In `SAML User ID Type`, select Assertion contains the Federation ID from the User object.
   - If your identity provider previously used the Salesforce username, communicate to them that they must use the Federation ID.
2. The **Entity ID** must be unique across your org and begin with `https`. You can't have two SAML configurations with the same **Entity ID** in one org. Specify whether you want to use the base domain (`https://saml.salesforce.com`) or the site URL (`https://acme.force.com/customers`) for the **Entity ID**. Share this information with your identity provider.

   Tip: Generally, use the site URL as the entity ID. If you are providing Salesforce to Salesforce services, you must specify the site URL.
3. In `SAML User ID Type`, select `Assertion contains the Federation ID from the User object`. If your identity provider previously used the Salesforce username, communicate to them that they must use the Federation ID.

Creating and Modifying Experience Cloud Users

The SAML assertion needs the following.

- A `Recipient` URL. This URL is the Site Login URL from the SAML Single Sign-On Settings detail page in your org. The URL is in the following form.

```
https://<community_URL>/login?so=<orgID>
```

For example, `Recipient="https://acme.force.com/customers/login?so=00DD0000000JsCM"` where `acme.force.com/customers` is the site home page and `00DD0000000JsCM` is the `Organization ID`.

If an Assertion Decryption Certificate has been uploaded to the org's SAML Single Sign-On Settings, include the certificate ID in the URL using the `sc` parameter. For example, `Recipient="https://acme.force.com/customers/login?so=00DD0000000JsCM&sc=0LE000000Dp"` where `0LE000000Dp` is the certificate ID.

- Salesforce attempts to match the `Federation ID` in the subject of the SAML assertion to the `User.FederationIdentifier` field of an existing user record. Salesforce can also use the attribute element to match, depending upon how the SAML Identity Location is defined in the SAML Single Sign-On Settings.
  1. If a matching user record is found, Salesforce uses the attributes in the SAML assertion to update the specified fields.
  2. If a user with a matching user record isn't found, then Salesforce searches the contacts for a match based on the `Contact ID` (`User.Contact`) or email (`Contact.Email`). `Contact.Email` and `Contact.LastName` are both required properties when `User.Contact` isn't specified, but matching is only based on `Contact.Email` when both properties exist.
     i. If a matching contact record is found, Salesforce uses the attributes in the SAML assertion to update the specified contact fields and then inserts a new user record.
     ii. If a matching contact record isn't found, then Salesforce searches the accounts for a match based on the `Contact.Account` or `Account.AccountNumber` specified in the SAML assertion. `Account.AccountNumber` and `Account.Name` are both required properties when `Contact.Account` isn't specified, but matching is only based on `Account.AccountNumber` when both properties exist.
        i. If a matching account record is found, Salesforce inserts a new user record and updates the account records based on the attributes provided in the SAML assertion.
        ii. If a matching account record isn't found, Salesforce inserts new account, contact, and user records based on the attributes provided in the SAML assertion.

  If the user account is inactive, the user account is updated but left inactive unless `User.IsActive` in the JIT assertion is set to true. If there is no user account with that `Federation ID`, the system creates a user.
- If the site doesn't have self-registration enabled and a default new user profile and role aren't specified, the `User.ProfileId` field must contain a valid profile name or ID.

Note: Salesforce also supports custom fields on the User object in the SAML assertion. Any attribute in the assertion that starts with `User` is parsed as a custom field. For example, the attribute `User.NumberOfProductsBought__c` in the assertion is placed into the field `NumberOfProductsBought` for the provisioned user. Custom fields are not supported for Accounts or Contacts.

Supported Fields for the Site SAML Assertion

To correctly identify which object to create in Salesforce, you must use a prefix. In the SAML assertion, use the `Account` prefix for all fields in the Account schema, for example, `Account.AccountId`. And use the `Contact` prefix for all fields in the Contact schema. In this example, the `Contact` prefix has been added to the `Email` field name.

```
<saml:Attribute
   Name="Contact.Email"
   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
      <saml:AttributeValue xsi:type="xs:anyType">testuser@123.org</saml:AttributeValue>
</saml:Attribute>
```

In addition to the standard User attributes supported for regular SAML JIT users, these Account attributes are also supported. For example, specifying an `Account.Phone` attribute in the assertion updates the account's Phone field on the corresponding Account object.

Name

AccountNumber

BillingCity

BillingCountry

BillingPostalCode

BillingState

BillingStreet

Owner
The Owner field on the Account object is Account.OwnerId in the API.

AnnualRevenue

Description

NumberOfEmployees

Fax

Industry

Ownership

Phone

Rating

ShippingAddress
The Shipping Address field is a compound field.

ShippingCity

ShippingCountry

ShippingPostalCode

ShippingState

ShippingStreet

Sic

TickerSymbol

Website

These Contact attributes are supported.

Account
This value is the Account Name field on the Contact object and Account.Name in the API.

Email

FirstName

LastName

Phone

CanAllowPortalSelfReg

AssistantName

AssistantPhone

Birthdate

Owner
This value is the Contact Owner field on the Contact object and Contact.OwnerId in the API.

Department

Description

DoNotCall

HasOptedOutOfEmail

Fax

HasOptedOutOfFax

HomePhone

LastCUUpdatetDate
This value is the Last Modified By field on the Contact object and Contact.LastModifiedDate in the API.

LeadSource

MailingAddress
The Mailing Address field is a compound field.

MailingCity

MailingCountry

MailingPostalCode

MailingState

MailingStreet

MobilePhone

Salutation

OtherAddress
The Other Address field is a compound field.

OtherCity

OtherCountry

OtherPostalCode

OtherState

OtherStreet

OtherPhone

Title

Just-in-Time Provisioning Errors

Following are the error codes and descriptions for Just-in-Time provisioning for SAML.

SAML errors are returned in the URL parameter, for example:

```
http://login.salesforce.com/identity/jit/saml-error.jsp?
ErrorCode=5&ErrorDescription=Unable+to+create+user&ErrorDetails=
INVALID_OR_NULL_FOR_RESTRICTED_PICKLIST+TimeZoneSidKey
```

Note: Salesforce redirects the user to a custom error URL if one is specified in your SAML configuration.

Error Messages

Code	Description	Error Details
1	Missing Federation Identifier	MISSING_FEDERATION_ID
2	Mis-matched Federation Identifier	MISMATCH_FEDERATION_ID
3	Invalid organization ID	INVALID_ORG_ID
4	Unable to acquire lock	USER_CREATION_FAILED_ON_UROG
5	Unable to create user	USER_CREATION_API_ERROR
6	Unable to establish admin context	ADMIN_CONTEXT_NOT_ESTABLISHED
8	Unrecognized custom field	UNRECOGNIZED_CUSTOM_FIELD
9	Unrecognized standard field	UNRECOGNIZED_STANDARD_FIELD
11	License limit exceeded	LICENSE_LIMIT_EXCEEDED
12	Federation ID and username do not match	MISMATCH_FEDERATION_ID_AND_USERNAME_ATTRS
13	Unsupported provision API version	UNSUPPORTED_VERSION
14	Username change isn't allowed	USER_NAME_CHANGE_NOT_ALLOWED
15	Custom field type isn't supported	UNSUPPORTED_CUSTOM_FIELD_TYPE
16	Unable to map a unique profile ID for the given profile name	PROFILE_NAME_LOOKUP_ERROR

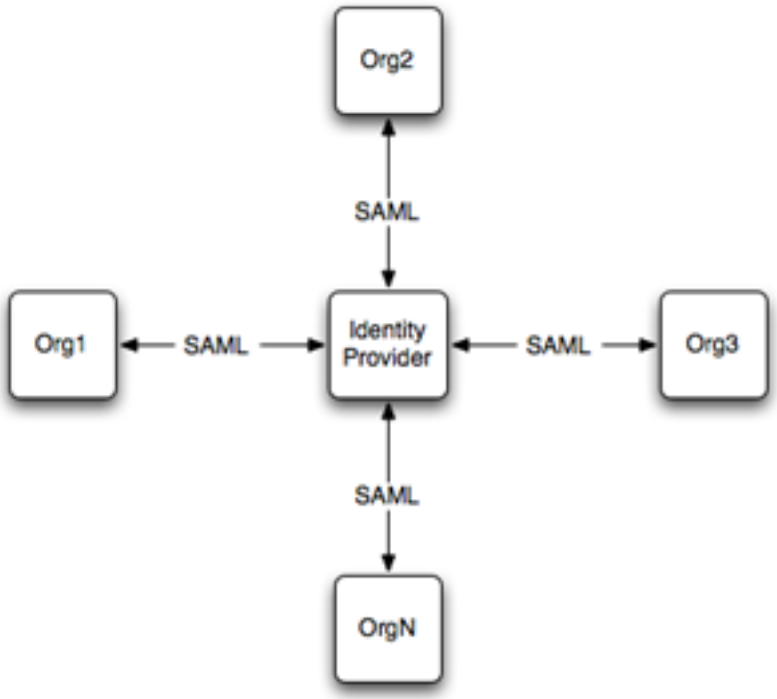
Code	Description	Error Details
17	Unable to map a unique role ID for the given role name	ROLE_NAME_LOOKUP_ERROR
18	Invalid account	INVALID_ACCOUNT_ID
19	Missing account name	MISSING_ACCOUNT_NAME
20	Missing account number	MISSING_ACCOUNT_NUMBER
22	Unable to create account	ACCOUNT_CREATION_API_ERROR
23	Invalid contact	INVALID_CONTACT
24	Missing contact email	MISSING_CONTACT_EMAIL
25	Missing contact last name	MISSING_CONTACT_LAST_NAME
26	Unable to create contact	CONTACT_CREATION_API_ERROR
27	Multiple matching contacts found	MULTIPLE_CONTACTS_FOUND
28	Multiple matching accounts found	MULTIPLE_ACCOUNTS_FOUND
30	Invalid account owner	INVALID_ACCOUNT_OWNER
31	Invalid portal profile	INVALID_PORTAL_PROFILE
32	Account change is not allowed	ACCOUNT_CHANGE_NOT_ALLOWED
33	Unable to update account	ACCOUNT_UPDATE_FAILED
34	Unable to update contact	CONTACT_UPDATE_FAILED
35	Invalid standard account field value	INVALID_STANDARD_ACCOUNT_FIELD_VALUE
36	Contact change not allowed	CONTACT_CHANGE_NOT_ALLOWED
37	Invalid portal role	INVALID_PORTAL_ROLE
38	Unable to update portal role	CANNOT_UPDATE_PORTAL_ROLE
39	Invalid SAML JIT Handler class	INVALID_JIT_HANDLER
40	Invalid execution user	INVALID_EXECUTION_USER
41	Execution error	APEX_EXECUTION_ERROR
42	Updating a contact with Person Account isn't supported	UNSUPPORTED_CONTACT_PERSONACCT_UPDATE

CONFIGURE SSO ACROSS MULTIPLE SALESFORCE ORGS

Let your users log in across multiple Salesforce orgs using single sign-on (SSO) credentials. With SSO, you can validate user credentials against a corporate database or other app rather than managing separate passwords for each Salesforce org.

Enterprises often deploy more than one Salesforce org. Unless you implement SSO, users that access different orgs must reauthenticate with each org. Removing this extra login step makes it more convenient for users and enhances security because it's easier for users to maintain and use a single, strong password.

SSO follows a hub-and-spoke architecture. At the center is a centralized authentication hub, the identity provider. The identity provider validates credentials and asserts the user's identity to the spokes—Salesforce orgs that are the service providers. The org that is the identity provider generates SAML assertions that follow the SAML 2.0 standard for SSO.



Salesforce supports both identity provider–initiated and service provider–initiated logins.

Set Up SSO from Salesforce to Salesforce

Configure one org as an identity provider and another org as a service provider.

1. Enable and deploy a My Domain subdomain in both Salesforce orgs.

   From Setup, in the Quick Find box, enter *My Domain*, and then select **My Domain**. Deploy the subdomain to the org's users.

   Warning: Deploying a domain on existing orgs can impact user bookmarks. Make sure that users are aware of this possibility before you deploy the subdomain on existing production orgs.

2. Set up one Salesforce org as the identity provider.

a. By default, Salesforce enables your org as an identity provider when you create the My Domain subdomain. To verify that your org is enabled as an identity provider, from Setup, in the Quick Find box, enter *Identity Provider*, and select **Identity Provider**. If the org isn't configured as an identity provider, click **Enable Identity Provider**.

b. Download a certificate from your identity provider. Your service provider uses a certificate to establish trust in the identity provider.

If you previously haven't generated a certificate and key pair, when Salesforce enables an identity provider, it creates one. Optionally, you can pick an existing self-signed certificate or use a CA-signed certificate.

c. Copy the **Salesforce Identity** URL listed under SAML Metadata Discovery Endpoints on the Identity Provider page. You can use this URL in the next step to import SAML metadata when you configure SAML SSO settings on the service provider. Alternatively, to capture metadata in an XML file, click **Download Metadata**.

3. Set up a service provider org.

Import metadata from a URL for the identity provider org. Alternatively, enter the SAML 2.0 settings manually, or import a metadata file with the settings.

a. From Setup, in the Quick Find box, enter *Single Sign-On Settings*, then select **Single Sign-On Settings**.

b. Click **Edit**. To reveal the SAML SSO settings, select **SAML Enabled**. Save the change.

c. Under SAML Single Sign-On Settings, click the appropriate button to create a configuration.

- **New**—Specify all settings manually.
- **New from Metadata File**—Import SAML 2.0 settings from an XML file from your identity provider. This option reads the XML file and uses it to complete as many settings as possible.

  Note: If your XML file contains information for more than one configuration, the first configuration that occurs in the XML file is used.

- **New from Metadata URL**—Import SAML 2.0 settings from a public URL listed under SAML Metadata Discovery Endpoints on the Identity Provider page. This option reads the XML file at this URL and uses it to complete as many settings as possible. The URL must be added to Remote Site Settings to access it from your Salesforce org.

d. Complete the SAML settings that describe the identity provider. The metadata options populate most settings, making it easier to set up an org or Experience Cloud site as a service provider. For example, the metadata populates the Issuer, Entity ID, and Identity Provider Login URL fields automatically with URLs for your identity provider.

To manage SSO effectively across multiple Salesforce orgs, specify federation ID as the SAML identity type.

Note: The structure of a Salesforce username is unique to each Salesforce org. Specifying federation ID as the SAML identity type allows an identity provider to map user identities across orgs. Each user has a common federation ID but a unique username across orgs.

After completing the settings, save them. From this page, copy the values from the Entity ID and Identity Provider Login URL fields. You need these values later when you define a connected app on the identity provider.

e. On the service provider, add the identity provider as an authentication service. From Setup, in the Quick Find box, enter *My Domain*, then select **My Domain**. Under Authentication Configuration, click **Edit**, and select the authentication service for your identity provider. Save the settings.

f. To define other orgs as service providers, repeat these steps.

4. On the identity provider org, create a Salesforce connected app.

Creating a connected app defines the service provider to the identity provider org.

a. Use the New Connected App wizard to define a connected app.

- In Lightning Experience, from Setup, in the Quick Find box, enter *App*, then select **App Manager**. Click **New Connected App**.
- In Salesforce Classic, from Setup, in the Quick Find box, enter *Apps*, then select **Apps**. On the page under Connected Apps, click **New**.

b. Remember the Entity ID and Login URL values that you saved? You need them now to configure settings for the connected app. Under Web App Settings, click **Enable SAML**. For **Entity ID**, paste in the Entity ID from the SAML SSO settings. For the **ACS URL**, enter the saved endpoint Login URL (for the org or site that's your service provider). To map user identities across multiple Salesforce orgs, specify **Federation ID** for the subject type. Save the settings.

c. To select the profiles and permission sets allowed to access this service provider, click **Manage** on the connected app page. Select **Manage Profiles** or **Manage Permission Sets**. Add the profiles or permission sets for users who can access this app.

d. When complete, Salesforce lists the new service provider on your identity provider's list of connected apps.

e. To define connected apps for other service provider orgs or sites, repeat these steps as needed.

To learn how to set up a Salesforce mobile app as a connected app, see this Trailhead.

5. Test the SSO implementation.

a. Create a test user in your identity provider org, and set the user's federation ID to a unique value. Make sure that you assign the user a profile that has been granted access to your service provider.

b. Create a test user in your service provider org, and set the user's federation ID to the same value as your test user in the identity provider. This action binds the two accounts together.

c. Log out from both orgs.

d. Enter the My Domain URL of the identity provider org into your browser, for example, `https://idp.mydomain.salesforce.com/`.

e. Log in to your identity provider org as the test user.

f. Now, enter the URL of the service provider org into your browser, for example, `https://sp1.mydomain.salesforce.com/`.

You're redirected to the identity provider org. Because you're already authenticated, you're redirected back to your service provider org. Presto, you're logged in!

Configure SSO from an Org to a Site

Implementing SSO between an org and a site is much the same as configuring SSO between two Salesforce orgs. What's different is that you must specify endpoint URLs that point to the site.

- To set up a site as a service provider, use the Experience Cloud Site Identity URL under SAML Metadata Discovery Endpoints on the Identity Provider page. Upload the SAML metadata from this URL. Using the metadata populates the service provider's SAML SSO settings, including the Login URL that points to the site. When you define a connected app on the identity provider, specify this Login URL as the ACS URL.
- To set up a site as an identity provider, complete the Identity Provider Login URL on the SAML Single Sign-On Settings page in the service provider org. This URL is where Salesforce sends a SAML request to start a login sequence. Use the URL for the site as the Identity Provider Login URL, rather than the URL for the org's My Domain subdomain. For example, to set up a site as an identity provider, specify the Identity Provider Login URL with the HTTP redirect binding.

  `https://acme.force.com/customers/idp/endpoint/HttpRedirect`

  In contrast, to set up an org as an identity provider, use the org's My Domain subdomain.

```
https://acme.my.salesforce.com/idp/endpoint/HttpRedirect
```

SEE ALSO:

Salesforce Help: SAML Login Flows

Configuring SAML SSO for a Canvas App

Configuring SSO for Mobile and Desktop Apps Using SAML and OAuth

FAQs for Single Sign-On

Review these frequently asked questions (FAQs) to help you implement and troubleshoot single sign-on (SSO).

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Federated Authentication is available in: **All** Editions

Delegated Authentication is available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Customer Portals and partner portals aren't available in **Database.com**

What are the different ways that I can implement SSO?

You can configure your Salesforce org as an identity provider, a service provider, or both. For each of these use cases, you select the authentication protocol to use. Salesforce supports SSO with SAML and OpenID Connect. Salesforce also has preconfigured authentication providers that you can use to enable SSO with systems that have their own authentication protocols, like Facebook. For more information, see Single Sign-On Use Cases. To see a SAML SSO implementation where Salesforce is the identity provider, watch this video.

Where can I view SSO errors?

You can view login errors in the Login History report. From Setup, in the Quick Find box, enter `Login History`, and then select **Login History**.

Does SSO work outside my corporate firewall?

Yes. When users are outside the corporate firewall, they can use their network passwords to log in to Salesforce. Alternately, you can require users to connect to your corporate network before logging in.

Can I validate the SAML response sent by my identity provider?

Yes. After you configure SSO, you can access the SAML Validation page from Setup, by clicking **SAML Validation** on the Single Sign-On Settings page. If a user tries to log in to Salesforce and fails, the invalid SAML assertion is used to automatically populate the SAML Assertion Validator. On the SAML Validation page, if the SAML assertion isn't automatically populated, you can enter the XML or base64-encoded SAML response that you received from your service provider. Salesforce validates the response against the values provided during SSO setup, and provides detailed information about the response.

Can I configure a start page and logout page that are specific to my company?

Yes. See Customize SAML Start, Login, Logout, and Error Pages on page 9.

Can I test my SSO configuration before implementing it?

Use a Developer Edition account or a sandbox to develop and test the configuration first. To sign up for a free Developer Edition account, go to developer.salesforce.com.

Keep in mind that sandbox copies are made with SAML disabled. Configuration information is preserved, except for the Salesforce Login URL. Salesforce Login URL is updated to match your sandbox URL after you re-enable SAML, for example, `https://yourInstance.salesforce.com/`. To enable SAML in the sandbox, from Setup, in the Quick Find box, enter `Single Sign-On Settings`, then select **Single Sign-On Settings**. Then click **Edit**, and select **SAML Enabled**.

Can I prevent users from logging in with their Salesforce user credentials?

To prevent users from logging in with a Salesforce username and password, assign the Is Single Sign-On Enabled user permission. You can do this directly in profiles or for specific users via permission sets. See Disable Logins with Salesforce Credentials for SSO Users for more information.

Can I enable SSO for Salesforce admins?

We don't recommend enabling SSO for Salesforce admins, in case there's an outage or other problem with your org's SSO implementation. Instead, admins should log in directly to Salesforce using multi-factor authentication (MFA).

If you do enable SSO for admins, they must still be able to log in directly to your org to address problems. For example, consider a situation where your third-party SSO provider has a sustained outage. An admin can use the standard Salesforce login page to log in with their username and password, then disable SSO until the problem is resolved.

Access the standard login page by modifying the Salesforce URL. You can add `login` as a query string parameter, for example, `https://northerntrailoutfitters-dev-ed.my.salesforce.com/?login`. Or you can append `login=true` to the URL, for example, `https://northerntrailoutfitters-dev-ed.my.salesforce.com/?login=true`.

Configuring SSO for Mobile and Desktop Apps Using SAML and OAuth

Salesforce mobile and desktop clients, including the SalesforceA mobile app for administrators, can combine OAuth and SAML protocols for service provider–initiated single sign-on (SSO).

App Support for SSO

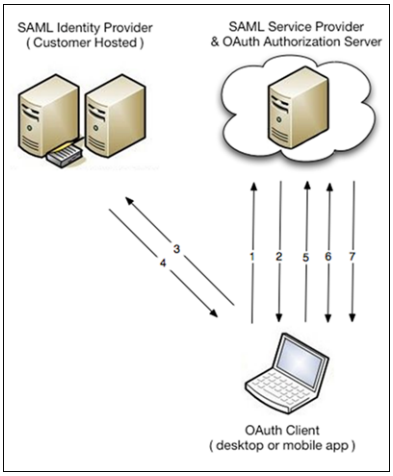
To authenticate mobile and desktop clients, a Salesforce org configured as a service provider can combine the OAuth and SAML protocols. OAuth allows users to connect applications to their accounts. SAML authenticates those connections. Using OAuth and SAML, mobile and desktop clients can take advantage of SSO integration in a way similar to web applications.

SSO integration is based on several core tenets.

- Developers are increasingly rewriting desktop and mobile applications to use OAuth to connect to user accounts. At runtime, users authenticate and authorize the app. After this initial step, a high-entropy (long, random) token is issued to the device. It is used instead of a password the next time the application is invoked. The token is unique to the user and application combination, and it is independently monitored, managed, and revoked.
- Many applications are now using web browsers to authenticate instead of native code. By using browsers, these applications can take advantage of SSO protocols written for the web. Previously, most applications were hard coded to ask users for credentials.
- Applications are separating authentication from authorization. By uncoupling these functions, an application adapts to change more easily. For instance, a client that normally prompts a user to log in using a web page from its servers can instead use SAML. It's also easy to implement adaptive risk-based authentication techniques. These measures remove the need for other, more cumbersome security measures, such as a Salesforce API token, which is a second credential that some applications use.
- Deployment must be simple and standard. Organizations that deploy SSO want to do so once and have it work everywhere.

Combining SAML and OAuth

By layering the SAML and OAuth protocols, mobile and desktop clients perform SSO using the process shown.



1. The OAuth client makes an authorization request to the hostname you specify. Using an embedded browser, the client asks the service provider for authorization. It does so using a custom URL that is your My Domain subdomain.
2. The authorization server detects that the client must authenticate and redirects the user to the SAML identity provider (IdP). The URL for the authorization server is passed via the RelayState parameter.
3. The user accesses the IdP, and the IdP performs authentication.
4. After the user is authenticated, the IdP sends back a SAML response. The browser transmits a response with a RelayState parameter. The response indicates that the client app is returning to the OAuth authorization server.
5. Salesforce processes the SAML assertion and logs the user in. The digital signature applied to the SAML response verifies that the message is from your system. At this point, Salesforce authenticates the user and redirects them to the authorization server.
6. After authentication, the client prompts the user to allow the client to connect to their account. The prompt is a simple web page that shows the user information about the client and what it's requesting.
7. If the user approves the application, it is issued a high-entropy token that the application uses to establish a session. Subsequent application use does not require the user to reenter credentials.

When layered with SAML, OAuth is much like any other bookmark or deep link. No additional development or deployment steps are necessary to enable SSO for the client app.

Configure Service Provider–Initiated SSO

Mobile or desktop apps that layer the SAML and OAuth protocols require two configuration steps.

1. Use the My Domain wizard to set up a subdomain. Deploying a subdomain improves the user experience and allows users to access deep links. This step is an easy way to prepare your org for service provider–initiated SSO from a mobile app or desktop client.
2. Point the client at the host that your subdomain represents. This step involves defining a new host connection in the app's settings.

Let's look at some examples that configure mobile and desktop clients for service provider–initiated SSO.

- Salesforce for iOS or Android on page 45
- SalesforceA for iOS on page 46
- SalesforceA for Android on page 46
- Desktop clients on page 46

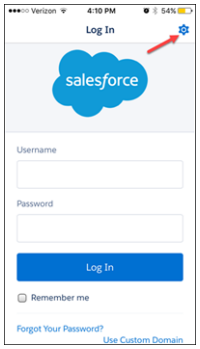
Various Salesforce and ISV applications support SSO using layered SAML and OAuth protocols. Because the combined protocol approach relies on open standards and public APIs, you can also use it for custom app development. If you are a developer that wants to take this approach and follow best practices, see Advice for Application Developers on page 47.

Salesforce for iOS or Android

The Salesforce mobile app works only with a Salesforce org configured for service provider–initiated SSO.

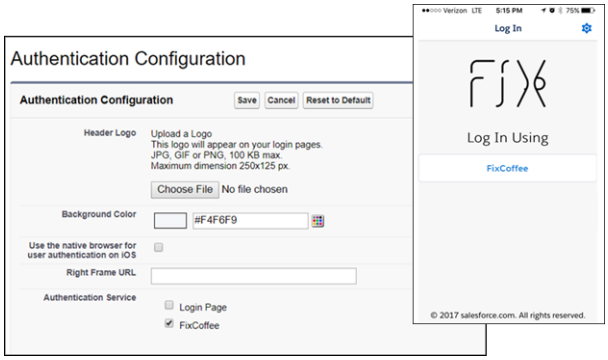
To set up Salesforce on either an iOS or Android device, configure your My Domain URL under the app's connection settings.

1. For example, on iOS, launch the app. At the top right of the Log In page, tap the gear icon.



2. To add a login connection, tap the **+** icon.
3. Enter the My Domain subdomain for your Salesforce org. Don't include `https://` in the URL. To save the new connection, tap **Done**.

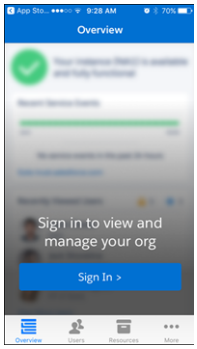
4. Select the connection you created, which redirects you to your org.
5. The login page appears. The login options depend on how you configured authentication services on your My Domain page. In this example, the login page is customized. The standard login authentication service is disabled, and the FixCoffee authentication service for the sample org is activated.



6. Enter your credentials, and log in to Salesforce.

SalesforceA for iOS

1. Launch the application, and tap Sign In.



2. Configure your My Domain subdomain as a new connection under the app's connection settings. To add a connection, tap the gear icon in the top right, and then the **+** icon.
3. Enter the My Domain subdomain for your Salesforce org. Don't include `https://` in the URL. To save the new connection, tap **Done**.

SalesforceA for Android

To set up SalesforceA on an Android device, configure a new connection that points to your subdomain.

1. Launch the SalesforceA app, and access the menu.
2. Tap **Change Server**, and then tap **Add Connection**.
3. Enter the My Domain subdomain, including `https://`. To apply the URL to the connection, tap **Apply**.
4. Choose the new connection, and tap **Apply**.

Desktop Clients

Several desktop clients, including Chatter Desktop, Salesforce for Outlook, and Data Loader, also layer the SAML and OAuth protocols. It's easy to configure these desktop apps for service provider–initiated SSO using a similar process.

1. Enable and deploy a My Domain subdomain in your Salesforce org.
2. Configure a new connection in the desktop client that points to your My Domain subdomain.

For example, to configure Data Loader as a desktop client, click **Use Custom Domain** on the login page. Enter the My Domain subdomain.

Advice for Deploying Applications

When deploying SSO for devices, consider the following best practices.

- Confirm that service provider–initiated SSO is working properly. Verify SSO using a desktop browser before trying it on a mobile device. Some deployments have difficulty properly propagating the RelayState parameter through the SAML request and response sequence. Verify that the IdP endpoints are correct, the RelayState's URL encoding is maintained, and the value sent to the IdP is echoed back to Salesforce.

  Note: The returned value must exactly match what's sent.

- Clearly communicate your My Domain URL to users and provide client-specific instructions. Users bear the responsibility of properly configuring apps to point to the right URL. Make sure that you educate users on the proper configuration steps.
- Consider the impact of IP restrictions on mobile devices. VPN and BES servers often help to alleviate issues. To circumvent IP restrictions, some mobile clients send activation emails.
- Assess the design of your identity provider's login page. Both the size and performance of mobile client pages can impact a user's experience. While Salesforce login services can dynamically adapt to various devices, when hosting your own identity provider, consider the size and loading speed of your login page. Consider implementing user agents that detect less-capable devices, and tailor or simplify authentication interfaces. This precaution is especially important when authenticating users on older device types.

Advice for Application Developers

When building OAuth-enabled applications, consider these best practices.

- Allow users to specify their My Domain URL. When deploying an application, it's important that users can specify the login service for authentication. Best practice is to allow a user to choose between production, sandbox (or test), and custom orgs. To use SSO, users must be able to specify a custom host connection. Also, consider techniques that simplify configuration, such as allowing an enterprise IT group to centrally manage user configurations.
- Consider the size of the login window. Given the wide variety of identity providers, it is difficult to predict the look and feel of the page presented when users log in using SAML. While some platforms, such as Android and iOS, can gracefully adapt to different page sizes, other platforms are less flexible. In these cases, consider using a reasonably large authentication interface, or allow the user to resize or scroll the interface.

Delegated Authentication

Another option to authenticate users on mobile and desktop devices is delegated authentication. Set up delegated authentication so users can log in to your Salesforce org with credentials managed by an external authentication method wrapped in a web service. When a user tries to log in to your org, Salesforce calls this web service to validate the user credentials. With delegated authentication, Salesforce has no control over the passwords used to log in to your org. Instead, the external authentication method controls user passwords and associated policies.

Enable the Is Single Sign-On Enabled user permission to hand control of user passwords to the external authentication method. Salesforce no longer manages the policies for user passwords, such as when passwords expire or the required minimum length. Instead, the delegated authentication endpoint's service enforces password policies, if any.

Salesforce uses this process for delegated authentication.

1. When a user tries to log in—either online or using the API—Salesforce tries to validate the username and checks the user's permissions and access settings.
2. If the Is Single Sign-On Enabled user permission is enabled, Salesforce calls to the SOAP-based SSO web service to validate the username and password.

Note: Salesforce immediately disposes of the password without storing, logging, or viewing it.

3. The web service call passes the username, password, and source IP to your SSO web service implementation, which Salesforce servers then access. The source IP is the address where the login request originated.
4. Your SSO web service implementation validates the passed information and returns either `true` or `false`.
5. When the response is `true`, the login process continues and the user is logged in to your org. When `false`, the user gets an error message that the username and password combination is invalid.

Note: With delegated authentication, a user can experience a slight delay when logging in while the user account becomes available in the org.

This process allows existing authentication systems to validate credentials, and it works with all mobile and desktop clients. After a delegated authentication service is built and enabled for a user, no additional configuration is required. The user can log in using the regular Salesforce login page, but your web service validates credentials.

Adapt Existing Delegated Authentication-Based SSO

Historically, many Salesforce customers have implemented SSO on top of delegated authentication, which follows this process.

1. Users start at an SSO page, perhaps on their corporate intranet.
2. Users log in locally to their own authentication system.
3. The SSO page generates a cryptographic token.
4. The Salesforce username and token are posted to the Salesforce login page. In addition, a startURL parameter is often included to indicate the user's starting page.
5. If the user profile has the Uses Single Sign-on permission enabled, Salesforce makes a web service call to a custom SSO service. Salesforce asks the service to validate the username and token.
6. The service validates the cryptographic token and returns either true or false. If the response is true, the login process continues, a new session is generated, and the user proceeds to the application.

SAML, an industry standard, has superseded delegated authentication. By using a SAML approach, companies can establish effective and standards-based SSO implementations.

If you have an existing SSO deployment using delegated authentication, you can apply the following technique with mobile or desktop clients. While delegated authentication deployments do not use SAML, they can be adapted to take advantage of these capabilities by using part of the SAML protocol. This change requires small adjustments to the code in your SSO service.

1. Configure My Domain and SAML for your org. You can configure dummy values for the SAML settings for everything except the Identity Provider Login URL. This setting must be a URL that receives the SAML authentication request from Salesforce, but it doesn't need to actually process the request.
2. A user requests a resource in the org, for example, `https://customer.my.salesforce.com/001/o`.
3. Salesforce notices that the user does not have a session for that org and sends a SAML authentication request to the org's SSO service. The request includes the RelayState parameter, for example, `/001/o`.
4. The SSO service receives both the SAMLRequest and RelayState parameters via an HTTP POST operation. The service ignores the SAMLRequest, but picks up the RelayState.
5. The SSO service authenticates the user and generates a cryptographic token.
6. The Salesforce username and token are posted to the Salesforce login page. In addition, a startURL parameter is echoed back with the value from the RelayState parameter.

7. If the user's profile has the Uses Single Sign-on permission enabled, Salesforce makes a web service call to the SSO service, asking it to validate the username and token.
8. The service validates the cryptographic token and returns either true or false. If the response is true, the login process continues, a new session is generated, and the user proceeds to the requested page.

To summarize, Salesforce sends the user request over SAML, but the request is sent back using delegated authentication. The RelayState parameter is transformed into the startURL parameter to redirect the user to the correct page. Using this technique, mobile and desktop clients that use SAML with Salesforce can also take advantage of SSO over delegated authentication.

SEE ALSO:

Configure SSO Across Multiple Salesforce Orgs

Salesforce Help: Configure SSO to Salesforce Using Microsoft AD FS as the Identity Provider

Salesforce Help: Salesforce for Outlook

Trailhead: Use the Salesforce Mobile App with Single Sign-On

Configuring SAML SSO for a Canvas App

Configuring SAML single sign-on (SSO) for a canvas app lets users easily access a new or existing application as a part of their Salesforce experience. A canvas app in one Salesforce org functions as an identity provider, authenticating another Salesforce org that's the service provider. The canvas app, hosted in the identity provider org, uses a signed request to reference a Visualforce page in the second org.

Create and Deploy My Domain Subdomain

In each org, use the Salesforce My Domain wizard to deploy a subdomain under my.salesforce.com.

If you haven't previously generated a certificate and key pair, setting up a My Domain subdomain also creates them. In a later step, you can use this certificate, or a CA-signed certificate or other self-signed certificate, to establish trust between your Salesforce orgs.

To provide information about your identity provider to your service provider, download the identity provider metadata.

1. From Setup, in the Quick Find box, enter `Identity Provider`, and then select **Identity Provider**.
2. Click **Download Metadata**. The metadata includes URLs and a self-signed certificate that you can use in a later step. If you want to copy the certificate from a separate file, click **Download Certificate**.

On the Identity Provider page, under SAML Metadata Discovery Endpoints, copy the Salesforce Identity URL. You can use this URL to import SAML metadata when you configure SAML SSO settings on the service provider.

Warning: Deploying a domain on existing orgs can impact user bookmarks. Make sure that your users are aware of this possibility before you deploy the subdomain on existing production orgs.

Configure SAML Settings on the Service Provider

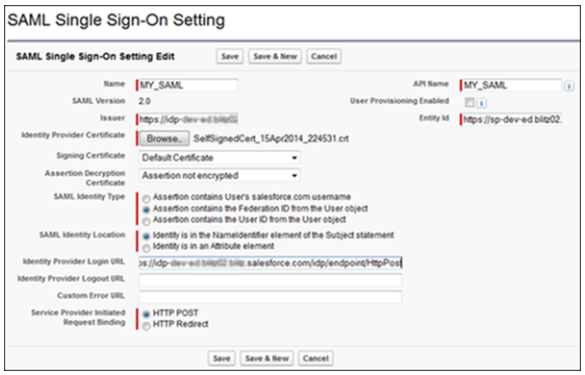
1. From Setup, in the Quick Find box, enter `Single Sign-On Settings`, and then select **Single Sign-On Settings**.
2. Click **Edit**. To reveal the SAML SSO settings, select **SAML Enabled**. Save your changes.
3. Under SAML Single Sign-On Settings, click one of the buttons to create a configuration.
   - **New**—All settings are manually specified.
   - **New from Metadata File**—Imports SAML 2.0 settings from an XML file from your identity provider. This option uses the XML file to complete as many settings as possible.

Note: If your XML file contains information for more than one configuration, the first configuration that occurs in the XML file is used.

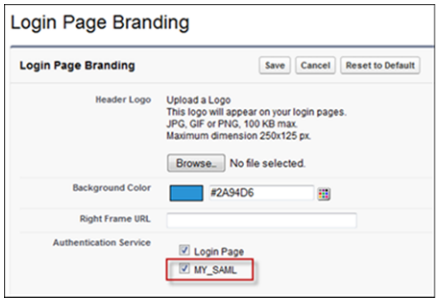
- **New from Metadata URL**—Imports SAML 2.0 settings from the public URL listed under SAML Metadata Discovery Endpoints on the Identity Provider page. This option uses the XML file to complete as many settings as possible. To access the URL from your Salesforce org, it must be added to Remote Site Settings.

Complete the SAML settings that describe the identity provider. When using metadata, most settings are automatically populated, making it easier to set up an org or Experience Cloud site as a service provider. For example, the metadata populates the Issuer, Entity ID, and Identity Provider Login URL fields with URLs for your identity provider.

Note: The structure of a Salesforce username is unique to each Salesforce org. To manage SSO across multiple Salesforce orgs, select **Federation ID** as the SAML identity type. This setting allows an identity provider to map user identities across orgs. Each user has a common federation ID across multiple orgs, but a unique username in each org.



4. Save your SAML settings.
5. Copy and save the Entity ID and the Login URL for the service provider org listed under Endpoints. You need these values when defining a connected app for the service provider.
6. Add the identity provider as an authentication service.
   a. From Setup, in the Quick Find box, enter `My Domain`, then select **My Domain**.
   b. Under Authentication Configuration, click **Edit**, and check the authentication service for your identity provider.
   c. Save the settings.



Create a Connected App in the Identity Provider

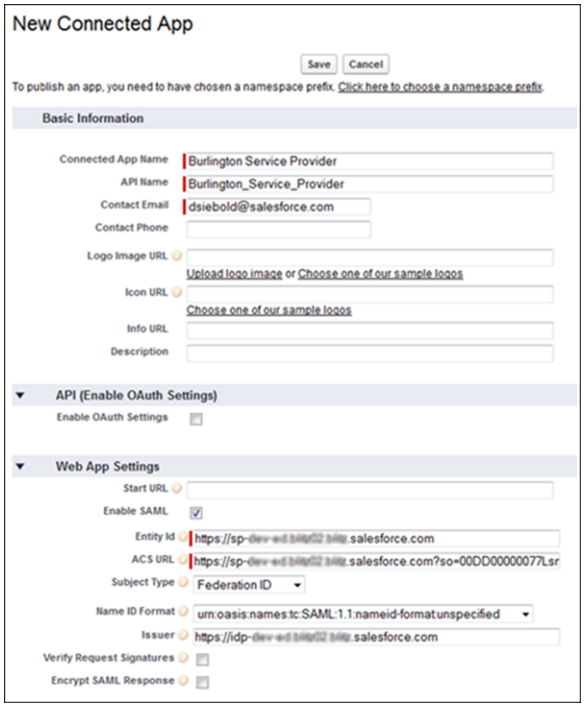
Creating a connected app defines the service provider to the identity provider.

1. Log in to the Salesforce org that's the identity provider.

2. Use the connected app wizard to define a connected app.
   - In Lightning Experience, from Setup, in the Quick Find box, enter *App*, and select **App Manager**. Click **New Connected App**.
   - In Salesforce Classic, from Setup, in the Quick Find box, enter *Apps*, and select **Apps**. Under Connected Apps, click **New**.
3. Under Web App Settings, click **Enable SAML** to enter the Entity ID and Login URL values that you saved.
   a. For Entity ID, enter the value from the SAML SSO settings.
   b. For ACS URL, enter the saved endpoint login URL of the org or site that's your service provider.
   c. To map user identities across multiple Salesforce orgs, select **Federation ID** for Subject Type. A federation ID is a unique value assigned to the user that lets you send authentication and authorization data between affiliated but unrelated web services.
   d. Save the settings.

   Note: It can take a few minutes for Salesforce to create the connected app.
4. To select the profiles and permission sets allowed to access this service provider, click **Manage** on the connected app page. Select **Manage Profiles** or **Manage Permission Sets**. Add the profiles or permission sets for users who can access this app.

   When complete, Salesforce lists the new service provider in your identity provider's list of connected apps.



Test the Connected App

Be sure that you're not logged in to any other SSO apps when you test your configuration. Otherwise, authentication fails, and you get an error that authentication can't proceed using the identity provider certificate.

1. Log in to your Salesforce org that's the identity provider.
2. Create a test user in your identity provider org, and set the user's federation ID to a unique value. Make sure that you assign the user a profile that was granted access to your service provider.

3. Create a test user in your service provider org. Set the user's federation ID to the same value as your test user in the identity provider org. This action binds the two accounts together.
4. Log out from both orgs.
5. In a browser, enter the My Domain URL of the identity provider org, for example, *https://idp.my.salesforce.com/*.
6. Log in to your identity provider org as the test user.
7. In a browser, enter the URL of the service provider org, for example, *https://sp.my.salesforce.com/*.

You're redirected to the identity provider org. Because you're already authenticated, you're redirected back to your service provider org and logged in.

Create an Identity User

Set up your canvas app, and configure it to use SSO. Create an identity user in the Salesforce identity provider org and then bind that user to the admin user of the service provider org.

1. Log in to your org that's the identity provider.
2. Create a user as an identity user. Be sure to enter a unique value for the federation ID.
3. Log in to your service provider org. To bind the identity user to the service provider administrator, set the federation ID for the administrator to the same value as the ID of the identity user.
4. Log out of both orgs.
5. In a browser, enter the domain URL of the identity provider and log in as the identity user.
6. In the browser, enter the domain URL of the service provider. A button appears on the login page with the name of your connected app. Click the button.

The browser redirects to the identity provider and then back to the service provider. You're logged in to the service provider as the administrator because the federation ID binds the accounts together.

Create a Visualforce Page in the Service Provider

1. Log in as the administrator to your org that's the service provider.
2. From Setup, in the Quick Find box, enter *Visualforce Pages*, then select **Visualforce Pages**.
3. Click **New**.
4. For Label, enter a label for your Visualforce page. For example, enter *ServiceProviderPage*.
5. Replace the code in the Visualforce Markup tab with this code sample.

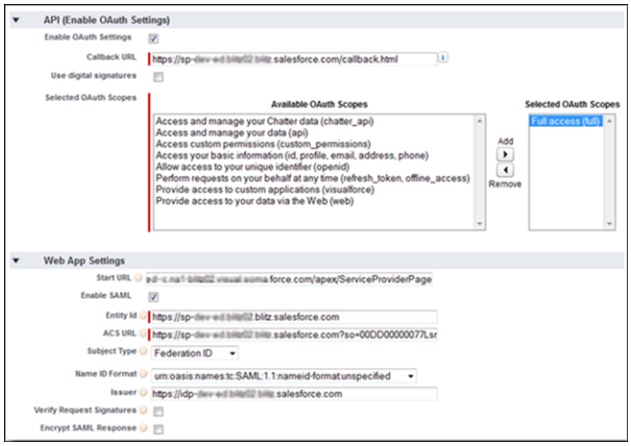
```
<apex:page showHeader="false">

<h1>Congratulations</h1>

My service provider Visualforce page

</apex:page>
```

6. Save the code.
7. Click **Preview** to review the Visualforce page.
8. Save the page URL, for example, *https://sp-<instance_name>.force.com/apex/ServiceProviderPage*.

9. From Setup, in the Quick Find box, enter *Visualforce Pages*, then select **Visualforce Pages**.
10. To give users access to the Visualforce page, click **Security** next to the page name. Add the Standard User to the list of enabled profiles for this page, and save the settings.

Update the Connected App in the Identity Provider

1. Log in to your Salesforce org that's the identity provider.
2. Modify the settings for the connected app defined previously.
   - In Lightning Experience, from Setup, in the Quick Find box, enter *App*, and select **Manage Connected Apps**.
   - In Salesforce Classic, from Setup, in the Quick Find box, enter *Apps*, and select **Apps**.
3. Next to your connected app, click **Edit**.
   a. For the Start URL, enter the URL of the Visualforce page you created in the service provider.
   b. Under API (Enable OAuth Settings):
      - Select **Enable OAuth Settings**.
      - For Callback URL, enter any URL. For example, *https://sp-<instance_name>.salesforce.com/callback.html*. You can use any value because you're using signed request authentication.
      - For Selected OAuth Scopes, select **Full Access (full)**.
   c. Under Canvas App Settings:
      - Select **Canvas**. This setting specifies that the connected app is a canvas app.
      - For Canvas App URL, enter the same URL that you entered for the Start URL.
      - For Access Method, select **Signed Request (POST)**.
      - For SAML Initiation Method, select **Identity Provider Initiated**. This setting indicates that the identity provider makes the initial request to start the SSO flow.
      - For Locations, add the Chatter tab to the Selected list of locations.
   d. Save the settings.



4. (Optional) Remove the login page from the service provider org.

If you selected **Service Provider Initiated** as the SAML Initiation Method when defining canvas app settings for your connected app, remove the login page from your service provider org. Doing so allows the service provider org to call the SAML SSO flow directly when it invokes the canvas app. In addition, the login page can't be contained in an iframe.

If you selected **Identity Provider Initiated** instead, you can keep the login page active for your service provider org.

Note: Log in to the service provider org as an administrator in a different browser. This way, if you run into issues, you can enable the login page.

a. To remove the login page from the service provider org, log in as the administrator to the org.

b. From Setup, in the Quick Find box, enter *My Domain*, then select **My Domain**.

c. Under Authentication Configuration, click **Edit**.

d. Deselect **Login Page** and save the settings.

Test the Canvas App

1. Log in as a standard user to your org that's the identity provider.
2. Click the **Chatter** tab. You see a link to the canvas app. The canvas app is actually a Visualforce page that resides in the service provider org.
3. Click the canvas app link. If the Visualforce page appears, the identity provider org authenticated the user into the service provider org.

Add Code to the Visualforce Page to Retrieve the Signed Request

When your canvas app appears, you know that SAML SSO authentication is working. However, as a result of the redirect, the initial signed request is no longer available. The signed request contains the context information and the required authentication token to communicate with the identity org. By adding some code to your Visualforce page, you can retrieve the signed request.

1. Log in as the administrator to your org that's the service provider. If you removed the login page, log in to the identity provider org as the identity user, and replace the browser URL with the service provider URL.
2. From Setup, in the Quick Find box, enter *Visualforce Pages*, then select **Visualforce Pages**.
3. Next to the name of your Visualforce page, click **Edit**.
4. Replace the code in the Visualforce Markup tab with this code.

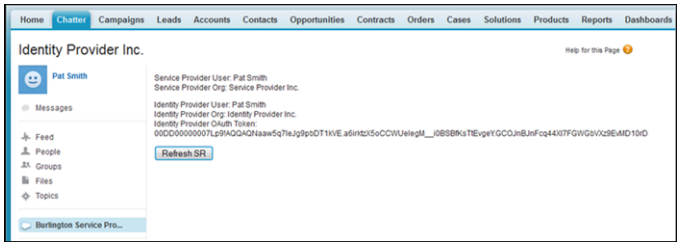
```
<apex:page showHeader="false">
<script type='text/javascript' src='/canvas/sdk/js/canvas-all.js'/>
<script>
  function refreshSR() {
      Sfdc.canvas.client.refreshSignedRequest(function(data) {
          if (data.status === 200) {
              var signedRequest = data.payload.response;
              var part = signedRequest.split('.')[1];
              var obj = JSON.parse(Sfdc.canvas.decode(part));
              updateDisplay(obj);
          }
      });
   }
  function updateDisplay(obj) {
      var oauth = document.getElementById('oauth');
      oauth.innerHTML = obj.client.oauthToken;
```

```
17          var user = document.getElementById('user');
18          user.innerHTML = obj.context.user.fullName;
19          var org = document.getElementById('org');
20          org.innerHTML = obj.context.organization.name;
21      }
22  </script>
23  <p/>
24  Service Provider User: {!$User.FirstName} {!$User.LastName}<br/>
25  Service Provider Org: {!$Organization.Name}<br/>
26  <p/>
27  Identity Provider User: <span id="user"></span><br/>
28  Identity Provider Org: <span id="org"></span><br/>
29  Identity Provider OAuth Token: <span id="oauth"></span><br/>
30  <p/>
31  <input id="refresh" type="button" value="Refresh SR" onclick="refreshSR();"/>
32  </apex:page>
```

5. Save the code.
6. Log out of both orgs.

Test the Updated Page

1. Log in as a standard user to the identity provider.
2. Click the **Chatter** tab. A link to the canvas app appears.



3. Click the link to the canvas app, and the Visualforce page appears. When you click **Refresh SR**, the page retrieves the signed request information for the user in the current org, which is the identity provider org.

You did it! You configured SAML SSO for a canvas app. Now you can give users seamless and secure access to your canvas apps.

SEE ALSO:

Salesforce Help: SAML Login Flows

Canvas Developer Guide

Configure SSO Across Multiple Salesforce Orgs

ENABLE SINGLE SIGN-ON FOR PORTALS

You can set up portals to use SAML single sign-on (SSO). However, customer portals and partner portals aren't available for new orgs as of the Summer '13 release. Use Experience Cloud sites instead.

1. Set up either a customer or partner portal.
2. In addition to the SAML sign-on information that you must gather and share with your identity provider, give your information provider the following information.
   - Org ID
   - Portal ID

   Add the following attributes to the SAML assertion sent from your identity provider.
   - `organization_id`
   - `portal_id`

   Get the values for org ID and portal ID from Setup.
   a. From Setup, in the `Quick Find` box, enter *`Company Information`*, then select **Company Information** and copy the ID located in `Salesforce Organization ID`.
   b. For the Customer Portal, from Setup, in the `Quick Find` box, enter *`Customer Portal Settings`*, then select **Customer Portal Settings**. Click the name of the Customer Portal, and then copy the ID located in `Portal ID`.

      For partner portals, from Setup, in the `Quick Find` box, enter *`Partners`*, then select **Settings**. Click the name of the partner portal, and copy the ID located in `Salesforce Portal ID`.

EDITIONS

Available in: Salesforce Classic (not available in all orgs)

Available in: **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS

To view the settings:
- View Setup and Configuration

To edit the settings:
- Customize Application

  AND

  Modify All Data

DELEGATED AUTHENTICATION

Delegated authentication is similar to single sign-on (SSO), but it offers a slightly different experience to users. With delegated authentication, one system relies on another system to validate user credentials. For example, you can configure your Salesforce org to rely on a Lightweight Directory Access Protocol (LDAP) server to validate credentials. Both SSO and delegated authentication enable users to log in to multiple apps with one set of credentials. However, with delegated authentication, users must log in to each app separately.

Set up delegated authentication so users can log in to your Salesforce org with credentials managed by an external authentication method wrapped in a web service. When a user tries to log in to your org, Salesforce calls this web service to validate the user credentials. With delegated authentication, Salesforce has no control over the passwords used to log in to your org. Instead, the external authentication method controls user passwords and associated policies.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

You can use any authentication method as long as you wrap it in a web service that Salesforce can consume. For example, you can use a Lightweight Directory Access Protocol (LDAP) server as your authentication method and wrap it in a SOAP-based web service. After you integrate the authentication backend with Salesforce, you can use Salesforce permissions to control which users log in with delegated authentication rather than with Salesforce credentials.

For example, your company uses an LDAP server for its employees. You want to use the LDAP server to authenticate users into Salesforce. You also want to use permissions on the user profile to determine whether users authenticate with LDAP or Salesforce. Specifically, you want users with standard profiles to log in with a password managed by the LDAP server, while system administrator profiles use a Salesforce password. So, you integrate your org with your LDAP server by wrapping the LDAP server in a SOAP-based web service. You create permissions so that only users with standard profiles use delegated authentication. Now, users with standard profiles enter a Salesforce username and the LDAP server handles their password. Users with system administrator profiles enter their Salesforce username and password.

Enable the Is Single Sign-On Enabled user permission to hand control of user passwords to the external authentication method. Salesforce no longer manages the policies for user passwords, such as when passwords expire or the required minimum length. Instead, the delegated authentication endpoint's service enforces password policies, if any.

Here's the process Salesforce uses to authenticate users with delegated authentication.

1. When a user tries to log in—either online or using the API—Salesforce tries to validate the username and checks the user's permissions and access settings.
2. If the Is Single Sign-On Enabled user permission is enabled, Salesforce calls to the SOAP-based SSO web service to validate the username and password.

   Note: Salesforce immediately disposes of the password without storing, logging, or viewing it.

3. The web service call passes the username, password, and source IP to your SSO web service implementation, which Salesforce servers then access. The source IP is the address where the login request originated.
4. Your SSO web service implementation validates the passed information and returns either `true` or `false`.
5. When the response is `true`, the login process continues and the user is logged in to your org. When `false`, the user gets an error message that the username and password combination is invalid.

Note: With delegated authentication, a user can experience a slight delay when logging in while the user account becomes available in the org.

More Help for Delegated Authentication

Configure Salesforce for Delegated Authentication

Set up delegated authentication for your Salesforce org so users can log in with credentials managed by an external authentication service. To configure Salesforce for delegated authentication, wrap your authentication method in a web service that Salesforce can consume. Then, use permissions to determine whether users log in with delegated authentication or with a Salesforce-managed password.

Here's the general process you use to set up delegated authentication:

1. Enable delegated authentication for your org.
2. Build your web service.
3. Specify your delegated authentication gateway URL.
4. Enable permissions.
5. (Optional) Record login attempts.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

USER PERMISSIONS

To view the settings:
- View Setup and Configuration

To edit the settings:
- Customize Application

  AND

  Modify All Data

Enable Delegated Authentication for Your Org

1. From Setup, in the Quick Find box, enter *Single Sign-On Settings*, then select **Single Sign-On Settings**.
2. Select **Disable login with Salesforce credentials**.

Build Your Web Service

1. In Salesforce, download the Delegated Authentication Web Services Description Language (WSDL) file.
   a. From Setup, in the Quick Find box, enter *API*, then select **API**.
   b. Click **Download Delegated Authentication WSDL**.

   The WSDL file describes the delegated authentication web service. Use the WSDL file to generate a server-side stub to which to add your delegated authentication implementation. For example, in the WSDL2Java tool from Apache Axis, use the `--server-side` switch. With the .NET wsdl.exe tool, use the `/server` switch.

2. Add a link to your corporate intranet or other internal site. This link takes the user's credentials after they're validated and passes them through an HTTP POST to the Salesforce login page.

   Salesforce only uses the password field to pass it back to your company. To avoid passing your corporate passwords to or from Salesforce, pass another authentication token instead, such as a Kerberos ticket.

   When the Salesforce server passes the credentials back to you in the `Authenticate` message, verify them. Then the user can access your org.

Specify Your Delegated Authentication Gateway URL

1. From Setup, in the Quick Find box, enter *Single Sign-On*, then select **Single Sign-On Settings**.
2. Click **Edit**.

3. Enter the URL for the `Delegated Gateway URL`.

   For security reasons, Salesforce restricts outbound posts to any of the following.

   - 80, which accepts only HTTP connections
   - 443, which accepts only HTTPS connections
   - 1024–66535, which accepts HTTP or HTTPS connections

Enable Permissions

Important: If you enable delegated authentication, API and desktop client users can log in to Salesforce, unless they try to log in from outside a restricted IP range. Also, the SSO authority usually handles login lockout policies for users with the Is Single Sign-On Enabled permission. However, if the security token is enabled, your org's login lockout settings determine how many times users can try to log in with an invalid security token.

1. Enable the Is Single Sign-On Enabled permission for all the users you want to use delegated authentication. See User Permissions and Access.

(Optional) Force Callouts to the Delegated Authentication Endpoint

1. From Setup, in the Quick Find box, enter *`Single Sign-On`*, then select **Single Sign-On Settings**.
2. Click **Edit**.
3. Select **Force Delegated Authentication Callout**.

   This setting forces a callout to the delegated authentication endpoint with all login attempts, even attempts that fail within Salesforce due to restrictions.

SAMPLE DELEGATED AUTHENTICATION IMPLEMENTATIONS

Samples are available by downloading a ZIP file containing sample code for .NET.

The samples are written in C# and authenticate users against Active Directory. The first sample is a simple implementation of delegated authentication. The second is a more complex sample that demonstrates a single-sign-on solution with an authentication token. Both samples use Microsoft .NET v1.1 and were deployed using IIS6 on a Windows 2003 server. Use the included `makefile` to build the samples.

Sample 1

The example in `simple.asmx.cs` declares a new class, `SimpleAdAuth`, which is a web service with one method: `Authenticate`. Several attributes are declared on the method. The attributes control the formatting of the expected request and the generated response, and they set up the service to match the message definition in the WSDL. The implementation uses the passed credentials to try to connect to Active Directory via the LDAP provider. If it connects successfully, the credentials are good. Otherwise, the credentials are invalid.

Sample 2

This more-complex example generates and verifies an authentication token rather than a password. The bulk of the implementation is in the `sso.asmx.cs` file, which defines a `SingleSignOn` class that generates an authentication token and implements the authentication service to verify that token. The generated token consists of a token number, expiration timestamp, and username. All the data is then encrypted and signed.

The verification process verifies the signature, decrypts the token, checks that it has not expired, and checks that the token number has not been previously used. (The token number and expiration timestamp are used to prevent replay attacks.) The file `gotosfdc.aspx` is an ASPX page designed to be deployed or linked to from an intranet site. This approach forces the user's authentication, generates a new authentication token for the user, and finally POSTs that token to the Salesforce login page along with a username that is mapped from the local NT username. The Salesforce login process sends the authentication token back to the service, which verifies the token and logs in the user. The file `intranet.aspx` is a simple page that links to `gotosfdc.aspx` so that you can see this process in action.

SINGLE LOGOUT

With single logout (SLO), your users can log out from a single application and be automatically logged out from all connected apps.

Because users are logged out of all connected apps at once with a single logout, SLO can improve security and usability. For example, when Salesforce is the identity provider for a connected app, the user logs out from Salesforce and is automatically logged out of the connected app. When a user is logged in to Salesforce from a connected app with a third-party identity provider, the user logs out of Salesforce and is automatically logged out of the connected app.

To use SLO, the identity providers, service providers, and relying parties must be configured for single sign-on (SSO) and registered for SLO.

Salesforce currently supports front-channel SLO only, meaning that SLO redirects must occur in the same browser. Salesforce doesn't support SLO across different browsers. Your users are only logged out of their registered apps if they explicitly log out of a connected app through a browser. If a browser session expires, users aren't logged out of the other apps registered for SLO.

Salesforce supports the following protocols (SLO can be initiated from the identity provider or any of the connected apps):

- SAML SLO as an identity provider or service provider
- OpenID Connect SLO as an identity provider or relying party

EDITIONS

Available in: both Salesforce Classic and Lightning Experience

USER PERMISSIONS

To view the settings:

- View Setup and Configuration

To edit the settings:

- Customize Application

  AND

  Modify All Data

Session Index Support

Salesforce supports session index parameters in requests and responses with SAML SLO. When a user logs out of a connected app registered for SAML SLO, the session index parameter is required to identify which user session to end.

- As the identity provider, Salesforce generates and sends the session index parameter to the service provider during SAML SSO. Depending on the initiating provider, SAML SLO follows one of these processes.
  - If Salesforce initiates SLO, Salesforce sends the same session index parameter with the logout request to the service provider.
  - If the service provider initiates SLO, Salesforce sends the SAML SLO request to the other service providers participating in the current session. The other service providers post a logout response to Salesforce. Salesforce then returns the logout response to the initiating service provider.
- As the service provider, Salesforce receives and stores the session index parameter sent from the identity provider during SSO. If the identity provider initiates SLO, Salesforce sends a logout response. If Salesforce initiates the SLO, it sends the same session index parameter with the logout request to the identity provider.

  Note: If the identity provider sends more than one session index parameter, Salesforce stores only the first one that it receives. The session index parameter can't be more than 512 characters.

SLO Examples

- You configure Salesforce as the identity provider. When users log in to Salesforce, they can access connected apps without logging in to each app separately. When users log out from Salesforce or a configured service provider or relying party, they're logged out of all connected apps and services. You can accomplish this behavior in the following ways.
  - SAML SLO for which Salesforce is the identity provider, and registered SAML connected apps are the service providers.

    With session index support, Salesforce sends a session index parameter during SSO. If Salesforce initiates SLO, it resends the same session index parameter with the logout request. Or if the service provider initiates SLO, Salesforce returns the final logout response to the initiating service provider.
  - OpenID Connect SLO for which Salesforce is the identity provider, and registered OAuth connected apps are the relying parties.
- You configure Salesforce to use a third-party identity provider. The identity provider uses SAML or OpenID Connect to log in users to a Salesforce org. When users log out of the identity provider or the Salesforce session, they're logged out of both. You can accomplish this behavior in the following ways.
  - SAML SLO for which Salesforce is the service provider connected to a third-party SAML identity provider.

    With session index support, Salesforce receives and stores the session index parameter during SSO. If Salesforce initiates SLO, it resends the same session index parameter with the logout request. Or if the service provider initiates SLO, Salesforce sends the final logout response.
  - OpenID Connect SLO for which Salesforce is the relying party connected to a third-party OpenID Connect provider.

SLO Benefits

Implementing SLO brings several advantages to your org.

- Time savings—With SLO in place, users avoid manually logging out of connected apps, which saves time and reduces frustration.
- Increased security—Users don't have to remember to log out of connected apps. When they log out of Salesforce, they're also logged out of the connected apps. Even if a user leaves a desktop unattended, nobody can access these apps.

Configuration Help

SEE ALSO:

Configure SAML Settings for Single Logout When Salesforce Is the Service Provider

Configure SAML Settings for Single Logout When Salesforce Is the Identity Provider

Configure OpenID Connect Settings for Single Logout Where Salesforce Is the Relying Party

Configure OpenID Connect Settings for Single Logout Where Salesforce Is the OpenID Connect Provider

Configure SAML Settings for Single Logout When Salesforce Is the Service Provider

When Salesforce is the service provider connected to an external SAML identity provider, users log in to an identity provider. The identity provider uses SAML to log in users to the Salesforce org. When users log out of the identity provider or the Salesforce session, they're logged out of both.

Before you configure SAML single logout (SLO), review these tips:

- If the identity provider sends more than one session index parameter, Salesforce stores only the first one that it receives. The session index parameter can't be more than 512 characters.
- You can configure a custom logout URL and identity provider SLO URL. If Salesforce is the service provider and initiates the logout request, the identity provider can post a logout response to redirect the user back to Salesforce. In this scenario, Salesforce sends the user to the custom URL upon logout. If the identity provider doesn't redirect the user back to Salesforce in a logout response, the identity provider determines the SLO redirect URL upon logout.
- Some identity providers don't support logout initiated by the service provider. In this case, skip to step 6 in the setup. Users are logged out of Salesforce when initiated by the identity provider. However, logging out of Salesforce doesn't necessarily log the user out of the identity provider session.
- Salesforce currently supports front-channel SLO only, meaning that SLO redirects must occur in the same browser. Salesforce doesn't support SLO across different browsers.

Complete the following prerequisites:

- Enable My Domain.
- Get the Issuer URL from the identity provider. This URL uniquely identifies your SAML identity provider. SAML assertions sent to Salesforce must match this value exactly in the `<saml:Issuer>` attribute of SAML assertions.
- Get and save the certificate for validating signatures from the identity provider.
- Get the SLO URL from the identity provider.

1. In Setup, in the **Quick Find** box, enter *Single Sign-On Settings*, then select **Single Sign-On Settings**.
2. In SAML Single Sign-On Settings, select **New**.
3. On the SAML Single Sign-On Settings page, enter the SAML single sign-on information, and select **Single Logout Enabled**.
4. Select **Use Selected Request Signature Method for Single Logout** to apply the selected Request Signature Method (RSM) during SLO. If you don't select this option, the default RSM (RSA-SHA1) is applied.
5. For Identity Provider Single Logout URL, enter the SAML SLO endpoint of the identity provider.
   - When Salesforce initiates the logout, it sends the logout request with the session index parameter to this SLO endpoint.
   - When the identity provider initiates the logout request, Salesforce sends the logout response to this SLO endpoint.
   - In addition to the identity provider SLO URL, you can define a custom URL. If Salesforce initiates the logout, the identity provider can post a logout response to redirect the user back to Salesforce. In this scenario, Salesforce sends the user to the custom URL upon logout.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Federated Authentication is available in: **All** Editions

Delegated Authentication is available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Authentication Providers are available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

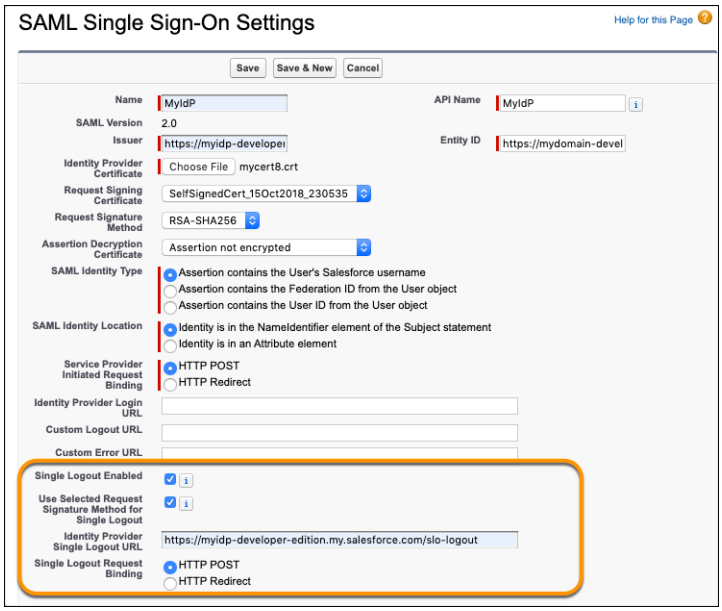
USER PERMISSIONS

To view the settings:
- View Setup and Configuration

To edit the settings:
- Customize Application

  AND

  Modify All Data

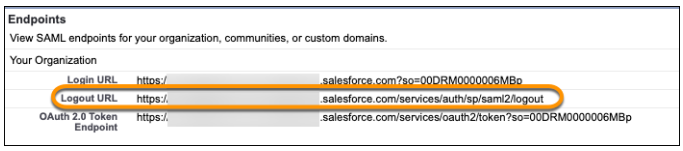


6. Select the HTTP binding type to use for SLO when initiated by the service provider. The binding type determines where to put the LogoutRequest or LogoutResponse in the SAML request. The value is base64 encoded.

   HTTP Redirect – in the query string, deflated.

   HTTP POST – in the POST body, not deflated.

7. Under Endpoints, for your organization logout URL, provide your identity provider with the Salesforce service provider SLO endpoint. The format for the endpoint is `https://<domain>.my.salesforce.com/services/auth/sp/saml2/logout`, where *<domain>* is your org's My Domain name.



   If the org is an Experience Cloud site, the logout URL for the site appears on the same page.

SEE ALSO:

Single Logout

Salesforce Help: Create Logout Event Triggers

Configure SAML Settings for Single Logout When Salesforce Is the Identity Provider

When Salesforce is the identity provider connected to an external SAML service provider, users log in to Salesforce. Salesforce uses SAML to log in users to the service provider through a connected app. When users log out of the service provider or the Salesforce session, they're logged out of both.

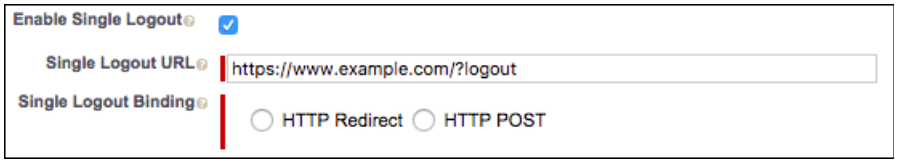
To use this feature:

- Enable My Domain.
- Make sure that the service provider supports SAML single logout (SLO).
- Get the SAML SLO endpoint from the service provider.
- Get the HTTP binding type from the service provider.

Review these considerations.

- This implementation uses connected apps. You can configure SLO when you create and edit a connected app as a developer and distribute it to other orgs. Or you can create and manage SLO for a connected app within your org as an administrator. When you're editing a connected app as a developer, your changes to the SLO configuration aren't propagated to the page. As you change settings through connected app management pages, manually copy settings to the app creation page, if desired.
- Salesforce currently supports front-channel SLO only, meaning that SLO redirects must occur in the same browser. Salesforce doesn't support SLO across different browsers.
- Salesforce generates and sends the session index parameter during single sign-on (SSO) and SLO.
- Some service providers don't support initiating SLO. In this case, skip to step 6 in the setup. Users are logged out of the service provider when initiated by Salesforce. But logging out of the service provider doesn't necessarily log the user out of Salesforce.

1. For an existing connected app: In Setup, in the **Quick Find** box, enter `apps`, then select **Manage Connected Apps**.
2. Next to the connected app that you want to configure for SLO, click **Edit**. You're now editing the connected app configuration even though the path is through Manage Connected Apps
3. Under SAML Service Provider Settings, select **Enable Single Logout**.



4. For Single Logout URL, enter the SAML SLO endpoint of the connected app service provider. The URL must start with `https://`.
   - When Salesforce initiates the logout, it sends the logout request with the session index parameter to this SLO endpoint.
   - When the service provider initiates the logout, Salesforce sends the logout response to this SLO endpoint.
5. Select the HTTP binding type for SLO. The binding type determines where to put the logout request or logout response in the SAML request. The value is base64 encoded. The service provider gives you this information.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Federated Authentication is available in: **All** Editions

Delegated Authentication is available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Authentication Providers are available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS

To view the settings:
- View Setup and Configuration

To edit the settings:
- Customize Application

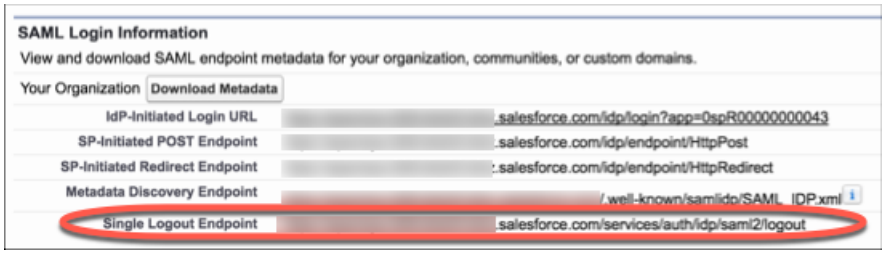
  AND

  Modify All Data

HTTP Redirect – in the query string, deflated.

HTTP POST – in the POST body, not deflated.

6. Provide your service provider with the Salesforce identity provider SLO endpoint. With this endpoint, the service provider can initiate SLO. Under SAML Login Information on the Connected App Detail page, it's listed in the Single Logout Endpoint and the SAML Metadata Discovery Endpoint. The format for the endpoint is `https://<domain>.my.salesforce.com/services/auth/idp/saml2/logout`, where `<domain>` is your org's My Domain name.



SEE ALSO:

Single Logout

Salesforce Help: Create Logout Event Triggers

Configure OpenID Connect Settings for Single Logout Where Salesforce Is the Relying Party

Configure single logout (SLO) when authentication providers use OpenID Connect to give users access to Salesforce as the relying party. Users log in to Salesforce through the authentication provider. When the users log out of Salesforce (or the authentication provider) session, they're automatically logged out of both.

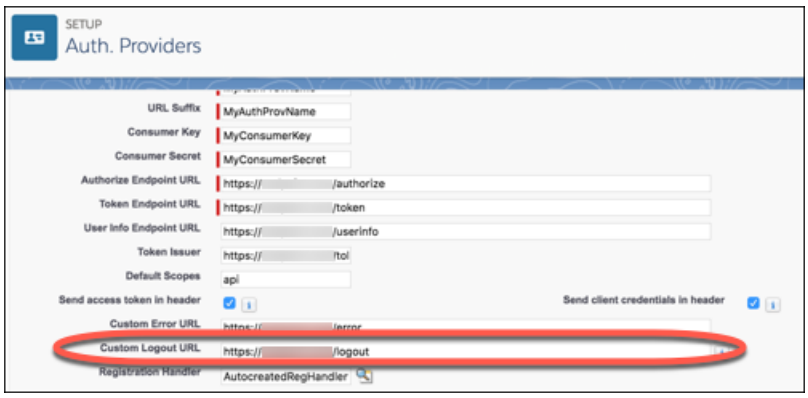
To use this feature:

- Enable My Domain.
- Make sure that the authentication provider supports OpenID Connect SLO.
- Set up the authentication provider.
- Get the OpenID Connect SLO logout endpoint from the authentication provider.

Review these considerations.

- Salesforce currently supports front-channel SLO only, meaning that SLO redirects must occur in the same browser. Salesforce doesn't support SLO across different browsers.
- Some authentication providers don't support logout initiated by the relying party. In this case, do only step 5. Users are able to log out of Salesforce when initiated by the authentication provider. But logging out of Salesforce doesn't necessarily log the user out of the authentication provider session.

1. In Setup, in the **Quick Find** box, enter `Auth. Providers`, then select **Auth. Providers**.
2. Next to the auth provider that you want to configure for SLO, click **Edit**.
3. Under **Auth. Provider Edit**, enter the logout endpoint from the authentication provider in **Custom Logout URL**. With this endpoint, Salesforce can initiate SLO. The Custom Logout URL must be an absolute URL and start with `http://` or `https://`.



4. Click **Save**.
5. Provide your authentication provider with the Salesforce SLO endpoint. With this endpoint, the authentication provider can initiate SLO. It's the **Single Logout URL** found under Salesforce Configuration on the Auth. Provider detail page. The format for the endpoint is `https://<domain>.my.salesforce.com/services/auth/rp/oidc/logout`, where `<domain>` is your org's My Domain name.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Federated Authentication is available in: **All** Editions

Delegated Authentication is available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Authentication Providers are available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

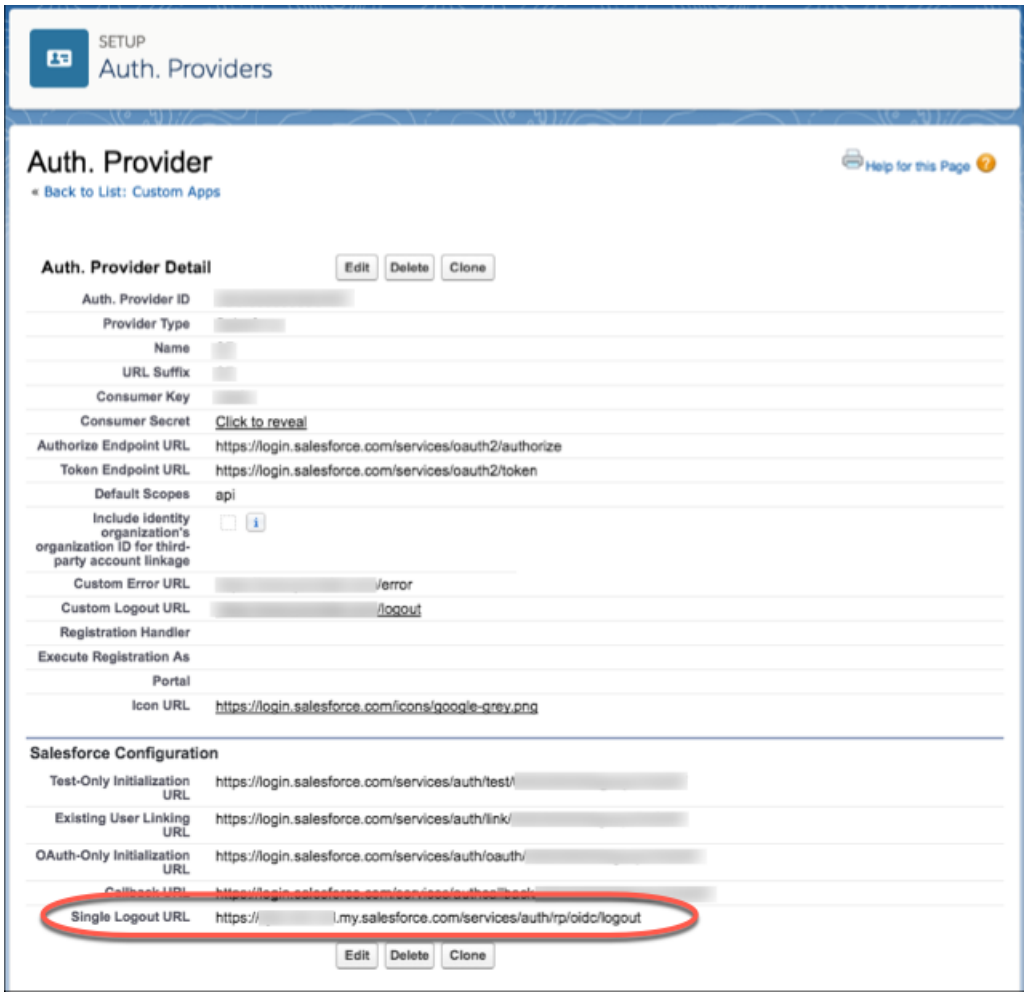
USER PERMISSIONS

To view the settings:
- View Setup and Configuration

To edit the settings:
- Customize Application

AND

Modify All Data



SEE ALSO:

- Single Logout
- *Salesforce Help*: Authentication Providers
- *Salesforce Help*: Create Logout Event Triggers

Configure OpenID Connect Settings for Single Logout Where Salesforce Is the OpenID Connect Provider

Configure single logout (SLO) when Salesforce provides authentication for users to access a relying provider using OpenID Connect. Users log in to Salesforce. Salesforce uses OpenID Connect to authenticate users for the relying party through a connected app. When the users log out of the relying party (or Salesforce) session, they're automatically logged out of both.

To use this feature:

- Enable My Domain.
- Make sure that the relying party supports OpenID Connect SLO.
- Get the OpenID Connect SLO logout endpoint from the relying party.

Review these considerations.

- Salesforce currently supports front-channel SLO only, meaning that SLO redirects must occur in the same browser. Salesforce doesn't support SLO across different browsers.
- After the initial creation of the connected app, changes to the SLO configuration for the connected app edit page don't automatically propagate to the manage page.
- When users initiate SLO from Salesforce, the redirect to the login page is delayed for about 10 seconds. This delay ensures that the user is also logged out of the relying party.

These steps edit an existing connected app. The fields are the same when you create or manage a connected app.

1. In Setup, in the **Quick Find** box, enter `apps`, then select **Manage Connected Apps**.
2. Next to the connected app that you want to configure for SLO, click **Edit**.
3. Under **OAuth Policies**, select **Enable Single Logout**.

4. For **Single Logout URL**, enter the OpenID Connect SLO endpoint of the connected app's relying party. This endpoint is where Salesforce sends a logout request when users log out of Salesforce. The relying party provides you with this endpoint. The Single Logout URL must be an absolute URL and start with `https://`.
5. Use the OpenID Connect Discovery Endpoint to provide your relying party with the Salesforce identity provider SLO endpoint. With this endpoint, the relying party can initiate SLO. It's found in `https://<domain>.my.salesforce.com/.well-known/openid-configuration`, where `<domain>` is your org's My Domain name. The format for the endpoint is `https://<domain>.my.salesforce.com/services/auth/idp/oidc/logout`, also where `<domain>` is your org's My Domain name.

EDITIONS

Available in: both Salesforce Classic (not available in all orgs) and Lightning Experience

Federated Authentication is available in: **All** Editions

Delegated Authentication is available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Authentication Providers are available in: **Professional**, **Enterprise**, **Performance**, **Unlimited**, and **Developer** Editions

USER PERMISSIONS

To view the settings:
- View Setup and Configuration

To edit the settings:
- Customize Application

  AND

  Modify All Data

SEE ALSO:

Single Logout

Salesforce Help: Create Logout Event Triggers

FREQUENTLY ASKED QUESTIONS

Review these frequently asked questions (FAQs) to help you implement and troubleshoot single sign-on (SSO).

What are the different ways that I can implement SSO?

You can configure your Salesforce org as an identity provider, a service provider, or both. For each of these use cases, you select the authentication protocol to use. Salesforce supports SSO with SAML and OpenID Connect. Salesforce also has preconfigured authentication providers that you can use to enable SSO with systems that have their own authentication protocols, like Facebook. For more information, see Single Sign-On Use Cases. To see a SAML SSO implementation where Salesforce is the identity provider, watch this video.

Where can I view SSO errors?

You can view login errors in the Login History report. From Setup, in the Quick Find box, enter *Login History*, and then select **Login History**.

Does SSO work outside my corporate firewall?

Yes. When users are outside the corporate firewall, they can use their network passwords to log in to Salesforce. Alternately, you can require users to connect to your corporate network before logging in.

Can I validate the SAML response sent by my identity provider?

Yes. After you configure SSO, you can access the SAML Validation page from Setup, by clicking **SAML Validation** on the Single Sign-On Settings page. If a user tries to log in to Salesforce and fails, the invalid SAML assertion is used to automatically populate the SAML Assertion Validator. On the SAML Validation page, if the SAML assertion is not automatically populated, you can enter the XML or base64-encoded SAML response that you received from your service provider. Salesforce validates the response against the values provided during SSO setup and provides detailed information about the response.

Can I configure a start page and logout page that are specific to my company?

Yes.

You can customize these pages for SAML SSO using external identity providers.

See Customize SAML Start, Login, Logout, and Error Pages on page 9.

Can I test my SSO configuration before implementing it?

Use a Developer Edition account or a sandbox to develop and test the configuration first. To sign up for a free Developer Edition account, go to developer.salesforce.com.

Keep in mind that sandbox copies are made with SAML disabled. Configuration information is preserved, except for the Salesforce Login URL. Salesforce Login URL is updated to match your sandbox URL, for example, `https://yourInstance.salesforce.com/`, after you re-enable SAML. To enable SAML in the sandbox, from Setup, in the Quick Find box, enter *Single Sign-On Settings*, then select **Single Sign-On Settings**. Then click **Edit**, and select **SAML Enabled**.

Can I prevent users from logging in with their Salesforce user credentials?

To prevent users from logging in with a Salesforce username and password, assign the Is Single Sign-On Enabled user permission. You can do this directly in profiles or for specific users via permission sets. See Disable Logins with Salesforce Credentials for SSO Users for more information.

Can I enable SSO for Salesforce admins?

We don't recommend enabling SSO for Salesforce admins, in case there's an outage or other problem with your org's SSO implementation. Instead, admins should log in directly to Salesforce using multi-factor authentication (MFA).

If you do enable SSO for admins, they must still be able to log in directly to your org to address problems. For example, consider a situation where your third-party SSO provider has a sustained outage. An admin can use the standard Salesforce login page to log in with their username and password, then disable SSO until the problem is resolved.

Access the standard login page by modifying the Salesforce URL. You can add `login` as a query string parameter, for example, `https://northerntrailoutfitters-dev-ed.my.salesforce.com/?login`. Or you can append `login=true` to the URL, for example, `https://northerntrailoutfitters-dev-ed.my.salesforce.com/?login=true`.

Are there any requirements for implementing my web service?

Your org's implementation of the web service must be accessible by Salesforce servers, so deploy the web service on a server in your DMZ. Remember to use your server's external DNS name when entering the delegated gateway URL in Single Sign-On Settings.

For security reasons, make your web service available by TLS. A certificate from a trusted provider, such as Verisign or Thawte, is required. For a list of trusted providers, contact Salesforce.

Are there special mapping considerations?

Map your org's internal usernames to your Salesforce usernames. If your org doesn't follow a standard mapping, extend your user database schema to include the Salesforce username as an attribute of a user account. Your authentication service can then use this attribute to map back to a user account.

Namespaces, element names, and capitalization must be exact in SOAP requests. Wherever possible, generate your server stub from the WSDL file to ensure accuracy.

How are passwords reset when delegated authentication has been implemented?

Password reset is disabled for delegated authentication because Salesforce no longer manages user passwords. Users who try to reset their passwords in Salesforce are directed to their Salesforce admin.

Where can I view delegated authentication errors?

Admins with the Modify All Data permission can view the 21 most recent login errors for your org. From Setup, in the Quick Find box, enter *`Delegated Authentication Error History`*, then select **Delegated Authentication Error History**. For each failed login, you can view the user's username, login time, and the error.

INDEX