OAuth and Connect REST API

Connect REST API uses OAuth to securely identify your application before connecting to Salesforce.

OAuth is an open protocol that allows secure authentication for access to a user's data, without handing out the user's username and password. It's often described as the valet key of software access. A valet key only allows access to certain features of your car. For example, you can't open the trunk or glove compartment using a valet key.

When you use OAuth, you avoid storing login credentials in your application. Instead, your application prompts the user to log in using a standard Salesforce page, which returns an access token to your application. Your application uses this token to access Connect REST API web services. Use Oauth in mobile apps and from a web page.

Connect REST API uses OAuth 2.0.

Connected Apps

To use Connect REST API, create a REST entry point in your organization. Salesforce defines this entry point as a connected app definition.

For an example of creating a connected app definition, see Step Two: Set Up Authorization.

Consider these items when creating and using a connected app.

- This table maps the labels used in the connected app definition to OAuth terms.

Connected App Label	OAuth Term	Description
`Consumer Key`	`client_id`	A unique identifier that identifies your application to Salesforce.
`Consumer Secret`	`client_secret`	A secret key associated with your application.
`Callback URL`	`redirect_uri`	A URL associated with your client application. In some contexts, the URL must be a real URL that the client's web browser is redirected to. In others, the URL isn't used. However, between your client application and the server (the connected app definition) the value must be the same. We suggest using a value that identifies the application, such as `http://mycomponent.myapp`.

- If you're developing a mobile app, use the following value for the `Callback URL` (`redirect_uri`) to avoid setting up an application server of your own.

```
https://login.instance_name/services/oauth2/success
```

  Don't use this value when developing a web application.
- Your connected app doesn't have to reside in the same org as your users. Use the connected app you create to sign in to any org.
- We recommend creating more than one connected app definition–one for testing and one for production. If you create only one, you must change the value of the `Callback URL` to reflect the location of your application code. In addition, we suggest creating more than one connected app if you're developing for different platforms, such as iOS and Android.

OAuth Basics

- OAuth grants access by client applications to resources owned by resource owners. In terms of OAuth, a resource is anything that you must keep secure. For Connect REST API, the resource we're concerned about protecting is all the data accessible through the API.
- It's up to a resource owner to grant access to resources. In our context, the resource owner is a combination of the admin (who administers users and the connected app) and the users (who log in and grant access to third-party applications).
- When an application wants to access a resource, it requests authorization from the resource owner. OAuth outlines various ways to grant an application ccess to a resource. These ways are referred to as grant types or flows. Different flows are suitable for different contexts.
- After a client application is authorized to access a resource, the client application receives an access token and a refresh token. The authorized client application must include the access token in any subsequent web service requests to identify itself. Access tokens have a limited lifetime. When an access token expires, the authorized client application can make a special request using the refresh token to obtain a new access token.

OAuth Flows

A user must be authenticated before accessing Salesforce. OAuth has multiple authentication flows. There are several steps in each authentication flow, as dictated by the OAuth standard and the type of application trying to access Salesforce. On successful authorization, the client application receives access and refresh tokens.

Salesforce supports these flows for use with Connect REST API.

- Web server flow
- User-agent flow

In addition, you can use the refresh token to get a new access token after your application has been authorized for access.

For detailed information about using OAuth with digital experiences, see Salesforce Help.

Revoke Access

After users have been granted access to a client application, they can revoke access by clicking **Revoke** in the Connected Apps section of their Personal Information page.

More Resources

Connect REST API shares some infrastructure with REST API. This information on authentication also applies to Connect REST API.

- *REST API Developer Guide*: Authorization Through Connected Apps and OAuth 2.0
- *Salesforce Help*: OAuth Authorization Flows
- *Salesforce Help*:Authorize Apps with OAuth

The authentication part of REST API is shared with Connect REST API. We suggest this library that supports REST API.

- JavaScript REST Toolkit

OAuth is a popular authentication standard. Client libraries for OAuth help smooth the development process for client applications. We suggest these client programming libraries.

- Ruby on Rails: OmniAuth
- Java: Apache Amber

- **Web Server OAuth Authentication Flow**
  Typically this flow is used by web applications that can confidentially store the client secret. A critical aspect of the web server flow is that the application must be able to protect the consumer secret.
- **User-Agent OAuth Authentication Flow**
  Typically this flow is used by mobile apps or applications that directly access the Connect REST API from JavaScript. In this flow, it's assumed that the client application can't be trusted to store client credentials, nor the user login credentials.
- **Tokens**
  As part of both authentication flows, you work with access tokens and refresh tokens.
- **Request Static Assets and Post Forms Directly from HTML**
  HTML pages that aren't hosted on Salesforce have had difficulty displaying user and group images and posting file attachments in forms. The reason is that URLs for these assets require authentication, which is provided with a session cookie when pages are hosted on Salesforce. Pages not hosted on Salesforce don't have access to the session cookie. Also, it isn't possible to pass an OAuth token from the HTML contexts `<img>`, `<a>`, and `<form>`.