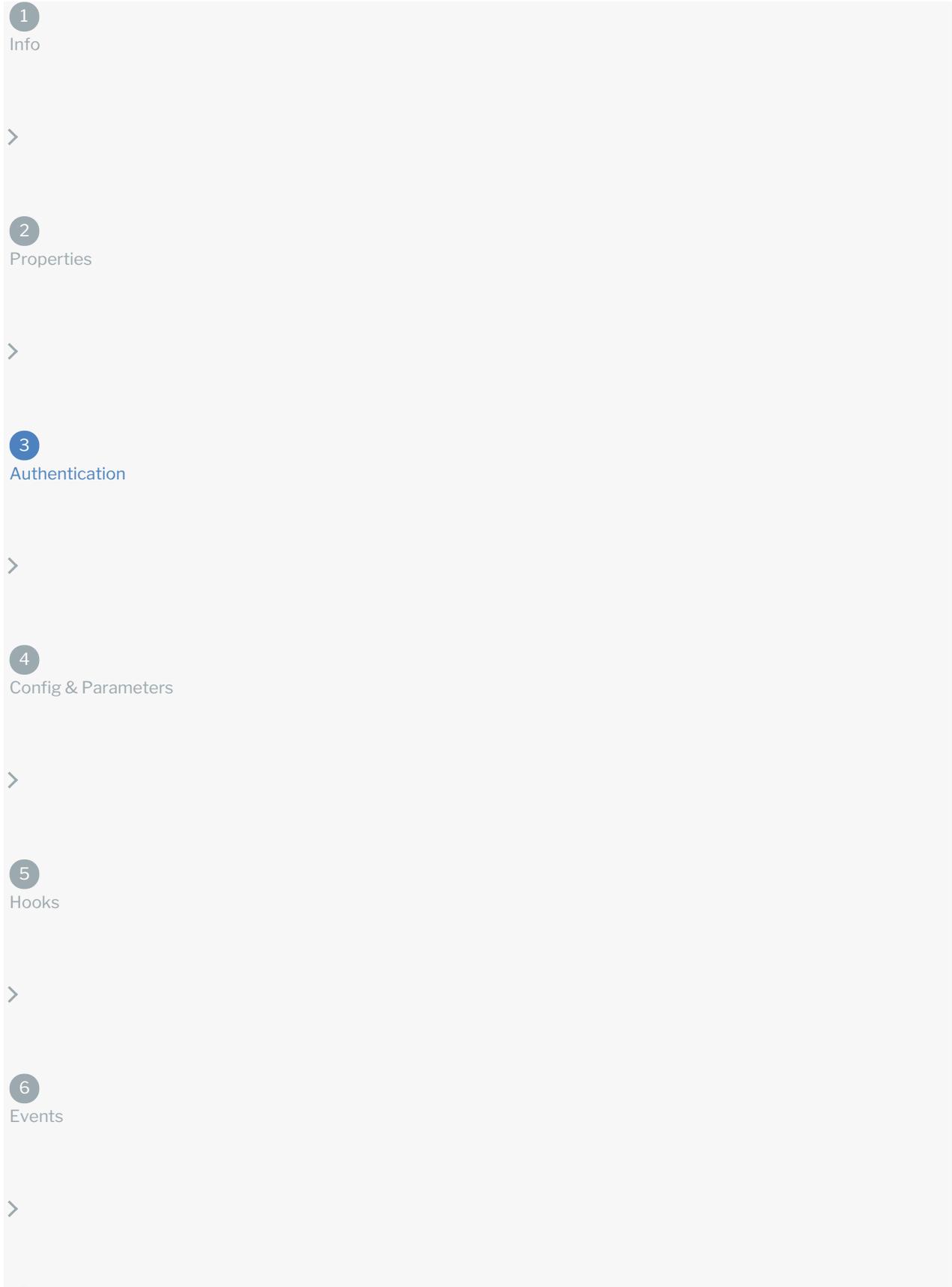


# 3: Setup Authentication

Last Modified on 09/22/2020 9:53 pm EDT



The information that you need to enter to set up authentication with the API provider differs depending on the authentication type. If the API provider requires complex authentication you can override the default information with configurations, parameters, and hooks.

Click the authentication type that you selected to see configuration instructions. If you selected Custom, you can skip directly to [Configuration and Parameters](#).

## Configure OAuth 2.0

The screenshot shows the SAP Connectors Setup page for OAuth 2.0 authentication. The 'Authentication' type is set to 'oauth2'. A table lists the following parameters:

Name	Key	Type	Default	Description	Required	Hide UI
OAuth Authorization UF	oauth.authorization_url	text+400	Default Value	Documentation OAuth Authorization URL	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
OAuth API Key	oauth.api.key	text+420	Default Value	Documentation OAuth API Key	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
OAuth API Secret	oauth.api.secret	text+420	Default Value	Documentation OAuth API Secret	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
OAuth Callback URL	oauth.callback_url	text+400	auth.cloudelements.io/c	Documentation OAuth Callback URL	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
OAuth Token URL	oauth.token_url	text+400	Default Value	Documentation OAuth Token URL	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
OAuth Scope	oauth.scope	text+64+	read	Documentation OAuth Scope	<input type="checkbox"/>	<input checked="" type="checkbox"/>
OAuth Token Refresh URL	oauth.token.refresh_url	text+400	Default Value	Documentation OAuth Token Refresh URL	<input type="checkbox"/>	<input checked="" type="checkbox"/>
OAuth Refresh Interval	oauth.user.refresh_interv	text+32+	3600	Documentation OAuth Refresh Interval (s)	<input type="checkbox"/>	<input checked="" type="checkbox"/>
OAuth Token Revoke URL	oauth.token.revoke_url	text+400	Default Value	Documentation OAuth Token Revoke URL	<input type="checkbox"/>	<input checked="" type="checkbox"/>

SAP provides the [properties](#) needed to support a standard OAuth 2.0 flow. Each API provider implements OAuth 2.0 differently so you might need to supplement the parameters with additional configuration. Before setting up the OAuth 2.0 information, you need to create a SAP app at the API provider. Use the default information from that app. When users authenticate through SAP, they will connect with that app.

The [OAuth 2.0](#) entry in the [About OAuth Authentication](#) section includes more details about the most common OAuth 2.0 authentication flow.

**Note:** Each of the parameters contributes to the configuration of the connector. You can use the configurations in parameters and hooks by referring to the Configuration Key shown in the table below.

To configure OAuth 2.0 connectors:

- If the Authentication section does not show OAuth 2.0 properties, select **oauth2** from **Authentication Type**.
- Complete the fields needed to enable users to grant SAP access to their account on behalf of your application:
  - OAuth API Key
  - OAuth API Secret
  - OAuth Callback URL
  - OAuth Authorization URL
  - OAuth Scope
- In **OAuth Token URL** enter the URL where SAP exchanges the grant code from the API provider for an access token.
- If the API provider expires tokens, complete the following fields:
  - OAuth Refresh Interval (s)
  - OAuth Token Refresh URL

- o OAuth Revoke Token URL

5. Unless you need to perform additional configuration, authenticate a test instance by clicking **Try it Out**.

## OAuth 2.0 Parameters

Name	Key	Description	Required
OAuth API Key	oauth.api.key	The default API Key used to authenticate with the API provider. Most API providers call this some variation of key or id, such as Key, API Key, or Client ID.	Y
OAuth API Secret	oauth.api.secret	The default API Secret used to authenticate with the API provider. Most API providers call this some variation of secret, such as Secret , API Secret, or Client Secret.	Y
OAuth Callback URL	oauth.callback.url	The URL that will receive the authorization code from the API provider used to authenticate a connector instance. For authentication through SAP, use <a href="https://auth.cloudelements.io/oauth">https://auth.cloudelements.io/oauth</a> .	Y
OAuth Authorization URL	oauth.authorization.url	The URL where a user authorizes the application to access their information at the API provider.	Y
OAuth Token URL	oauth.token.url	The URL where the application exchanges the authorization grant code or request token for an access token.	Y
OAuth Scope	oauth.scope	A comma separated list of the permissions that the user will authorize your integration to have.	N
OAuth Refresh Interval (s)	oauth.user.refresh_interval	If the access token expires, the time frame in seconds when SAP sends a request to the OAuth Token Refresh URL. The default is 3600, which is one hour.	N
OAuth Token Refresh URL	oauth.token.refresh_url	The URL to send a refresh request.	N
OAuth Revoke Token URL	oauth.token.revoke_url	The URL to send requests to revoke refresh or access tokens.	N

## Configure OAuth 1.0

The screenshot shows the SAP Connectors documentation interface for configuring an OAuth 1.0 authentication type. The 'Authentication' section is active, displaying a table of parameters with their keys, types, default values, descriptions, and required status.

Name	Key	Type	Default	Description	Required	Hide UI
OAuth Authorization UF	oauth.authorization.url	text:108	Default Value	Documentation OAuth Authorization URL	<input checked="" type="checkbox"/>	<input type="checkbox"/>
OAuth API Key	oauth.api.key	text:128	Default Value	Documentation OAuth API Key	<input checked="" type="checkbox"/>	<input type="checkbox"/>
OAuth API Secret	oauth.api.secret	text:128	Default Value	Documentation OAuth API Secret	<input checked="" type="checkbox"/>	<input type="checkbox"/>
OAuth Callback URL	oauth.callback.url	text:108	auth.cloudelements.io/c	Documentation OAuth Callback URL	<input checked="" type="checkbox"/>	<input type="checkbox"/>
OAuth Token URL	oauth.token.url	text:108	Default Value	Documentation OAuth Token URL	<input checked="" type="checkbox"/>	<input type="checkbox"/>
OAuth Scope	oauth.scope	text:64	read	Documentation OAuth Scope	<input type="checkbox"/>	<input type="checkbox"/>
OAuth Request URL	oauth.request.url	text:128	test.com	Documentation OAuth Request URL	<input checked="" type="checkbox"/>	<input type="checkbox"/>
OAuth User Secret	oauth.user.token.secret	text:128	Default Value	Documentation OAuth User Secret	<input checked="" type="checkbox"/>	<input type="checkbox"/>
OAuth Authorization Ty	oauth.request.authorizati	text:128	query	Documentation OAuth Authorization Type (Header or Query)	<input checked="" type="checkbox"/>	<input type="checkbox"/>

SAP provides the [parameters](#) required to support a standard OAuth 1.0 flow. Each API provider implements OAuth 1.0 differently so you might need to supplement the parameters with additional configuration.

The [OAuth 1.0](#) entry in the [About Authentication](#) section includes more details about the most common OAuth 1.0 authentication flow.

To configure OAuth 1.0 connectors:

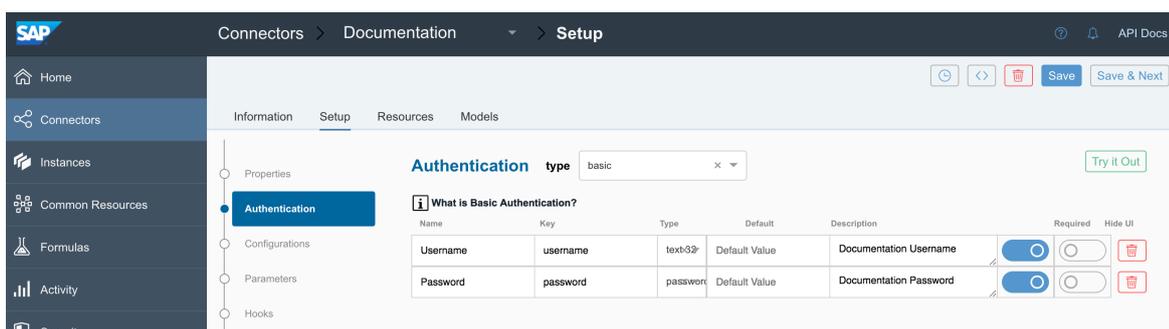
1. If the Authentication section does not show OAuth 1.0 properties, select **oauth1** from **Authentication Type**.
2. Complete the fields needed to get an OAuth Request Token to request user authorization
  - o OAuth API Key
  - o OAuth API Secret
  - o OAuth Request URL
  - o OAuth Callback URL
3. In **OAuth Authorization URL** enter the URL where SAP redirects the user to authorize access.
4. In **OAuth Token URL** enter the URL where SAP fetches an access token.
5. In **OAuth Authorization Type (Header or Query)** select how the request passes authorization information to the API provider.
6. In **OAuth Scope** enter a comma separated list of the permissions that the user will authorize your integration to have.
7. In **OAuth User Secret** enter
8. Unless you need to perform additional configuration, authenticate a test instance by clicking **Try it Out**.

## OAuth 1.0 Parameters

Name	Key	Description	Required
OAuth Request URL	oauth.request.url	The URL used to get an unauthorized request token.	Y
OAuth Callback URL	oauth.callback.url	The URL that will receive the authorization code from the API provider used to authenticate a connector instance.	Y
OAuth API Key	oauth.api.key	The default API Key used to authenticate with the API provider. Most API providers call this some variation of key or id, such as Key, API Key, or Client ID.	Y
OAuth API Secret	oauth.api.secret	The default API Secret used to authenticate with the API provider. Most API providers call this some variation of secret, such as Secret , API Secret, or Client Secret.	Y
OAuth Authorization URL	oauth.authorization.url	The URL where a user authorizes the application to access their information at the API provider.	Y
OAuth Token URL	oauth.token.url	The URL where the application exchanges the authorization grant code or request token for an access token.	Y
OAuth Authorization Type (Header or Query)	oauth.request.authorization.type	How the API provider receives authentication information, either in the header or as a query parameter.	Y
OAuth Scope	oauth.scope	A comma separated list of the permissions that the user will authorize your integration to have.	N
OAuth User Secret	oauth.user.token.secret	The user secret associated with the application authenticating with the API provider.	N

Name	Key	Description	Required
------	-----	-------------	----------

## Configure Basic Authentication

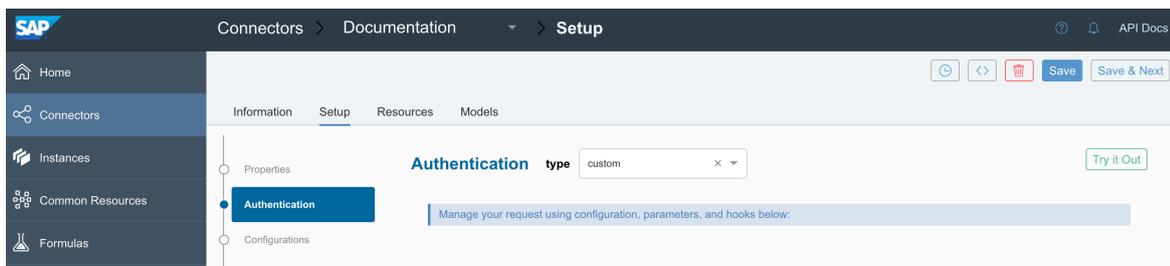


In Basic access authentication, you typically provide a user name and password. In some cases you also provide an API Key. When setting up a connector with Basic authentication we start you off with **Username** (key: `username` ) and **Password**(key: `password` ) configurations. If you need to add any other configurations like an API Key, do so in the Configuration step. API providers typically do not vary from the standard Basic authentication, so you should keep the default properties. If you do need to make changes, you can update the properties or delete unneeded configurations.

## Basic Authentication Parameters

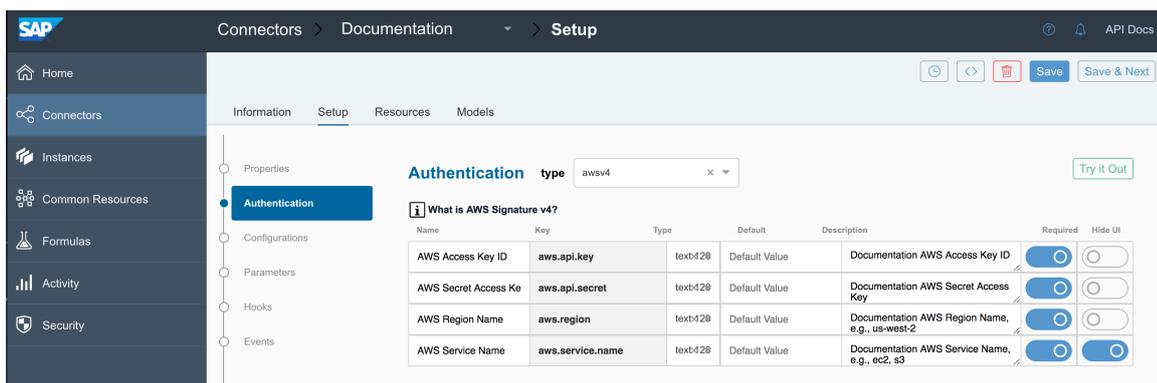
Parameter	Description	Required
Name	The visible name of the configuration as it appears in the SAP UI.	Y
Key	The internal unique identifier for the configuration. The configuration key value appears in the authentication JSON. You also refer to the configuration by configuration key in parameters and hooks.	Y
Type	The type of configuration, which can be any of the following:  Text Area, Text Text 32, Text 64, Text 100, Text 128 — Identifies configurations that accept free text strings.  true/false and yes/no — Identifies configurations that accept boolean inputs.  password — Identifies configurations intended to collect passwords. Passwords are masked in SAP.	Y
Required	Identifies whether the configuration is required. Switch on to force a user to provide data when authenticating. Configurations not required, but that show in SAP appear under Show Optional Fields.	Y
Hide on UI	Identifies whether the configuration appears in SAP. Switch on to show the configuration in SAP. The configuration appears to the user with the configuration name and the description as hover help text.	Y
Configuration Description	The description appears as hover help text. The text space available is limited, so write brief but useful descriptions.	Y
Default Value	The default value of the configuration.	N

## Configure Custom Authentication



To configure custom authentication information, use the [Configuration](#), [Parameters](#), and, if necessary, [Hooks](#) sections to construct the authentication information required by the API provider.

## Configure AWS Authentication



SAP provides default authentication fields for API providers that use [Amazon Web Services Signature Version 4](#) and [Version 2](#). API providers typically do not vary from the standard AWS authentications, so you should keep the default properties. If you do need to make changes, you can update the properties or delete unneeded configurations.

## AWS Authentication Parameters

Name	Description	Required
AWS Access Key ID	The ID associated with your AWS access key.	Y
AWS Secret Access Key	The secret key used in the Signature Version 2 signing process	Yes for version 2 only
AWS Region Name, e.g., us-west-2	The <a href="#">Amazon API Gateway</a> region name.	Yes for version 4 only

## Change the Authentication Type

If you selected the incorrect authentication type, or find that you need to change the authentication, select another authentication type from the **type** list in the Authentication heading.

## Test Authentication

After you set up authentication you can authenticate a connector instance to see if you can create a connection. If you chose Custom authentication or need to add additional configurations and parameters, you should set those up before testing authentication.

To test your authentication:

1. Click **Try it Out**.

2. On the authentication page, enter a name, and then complete any required configuration fields. These are the fields that you chose to appear on the UI.
3. Click **Create Instance**.

If the authentication succeeds, SAP creates an authenticated instance which you can use to test your authentication. You can also use it later when you test resources.

## About OAuth Authentication

OAuth is a common authentication protocol for REST APIs and SAP supports both [OAuth 2.0](#) and [OAuth 1.0](#). You need to set up more information for OAuth connectors than other authentication types. This section provides more details about how these authentication types work and how SAP uses the information you provide when building custom connectors.

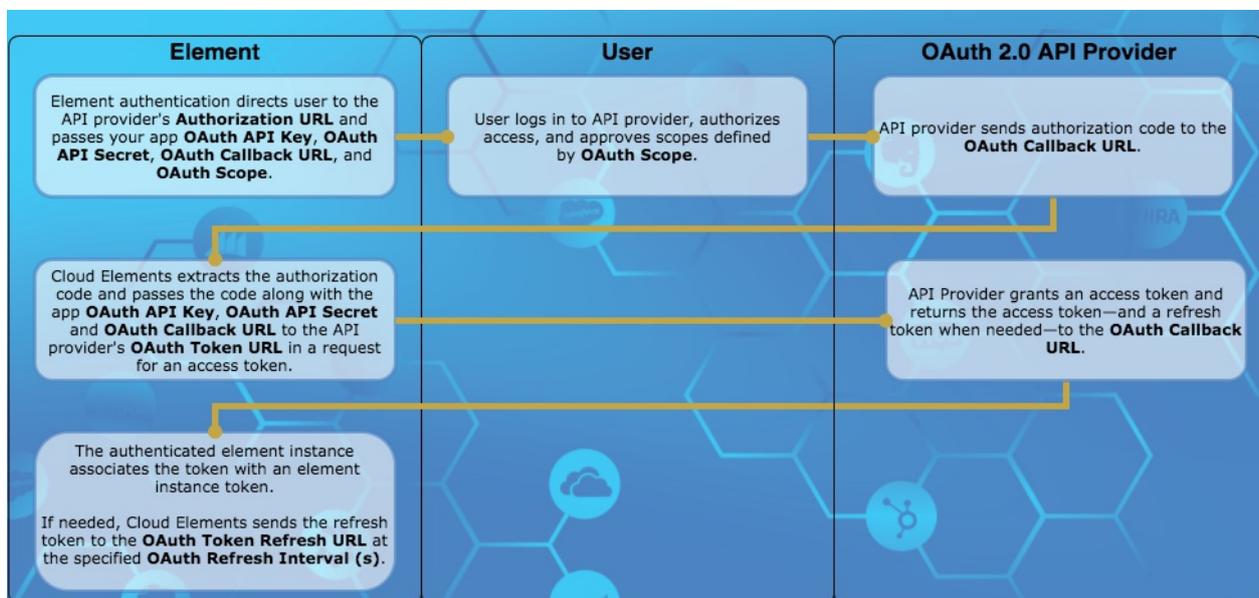
## OAuth 2.0

The OAuth 2.0 protocol lets external applications — your application or SAP — request authorization to access and update a users data without asking users for sensitive user names and passwords. OAuth 2.0 is flexible, but that also means that each API provider implements it differently.

This section describes how SAP supports the [Authorization Code Grant workflow](#) authentication flow. Other flows such as [Implicit Grant](#) and [Client Credentials Grant](#) offer variations that we also support. The Authorization Code Grant flow is sometimes called a three-legged authentication process:

1. SAP requests an API provider to have a user log in and grant access.
2. After the user logs in and grants permission, the API provider returns an authorization grant code.
3. SAP exchanges the authorization grant code for an access token.

### SAP Authorization Code Grant OAuth 2.0 Flow



The typical Authorization Code Grant flow as supported by SAP includes the following steps:

1. SAP requests authorization on behalf of your app by redirecting a user to the API provider's **Authorization URL**. SAP includes the following parameters as part of the request:
  - o **OAuth API Key**
  - o **OAuth API Secret**
  - o **OAuth Callback URL**
  - o **OAuth Scope** if required

2. The user logs in and grants access. If the request included **OAuth Scope**, the user must grant access to all scopes or none.
3. The API provider redirects the user back to the **OAuth Callback URL** with the following parameters:
  - A time-limited authorization grant `code` .
  - A `state` parameter that is typically the connector key.

**Note:** If the user denies authentication, the API provider returns `error` instead of the `code` parameter.

4. SAP makes a backchannel request to redeem the authorization grant code for an access token.
5. The API provider authenticates the connector instance and issues an access token. Some API providers also provide a refresh token and information about when the token expires.
6. SAP associates a connector instance token with the access token. You will use that connector instance token in future requests.
7. If the tokens expire, SAP sends the refresh token to the **OAuth Token Refresh URL** at the specified **OAuth Refresh Interval (s)**.
8. The API provider responds with a new access token and refresh token to be used at the next refresh.

## OAuth 2.0 Prerequisites

Before you can authenticate a connector instance, you must register your application with the API provider. Each registration is assigned some form of an API key (client id, client key, etc.) and secret. In SAP, these are the **OAuth API Key** and **OAuth API Secret**. You also usually need to provide a redirect URI which should match what you include in the **OAuth Callback URL**. If you want users to authenticate through SAP as opposed to programmatically through APIs in your application, use `https://auth.cloudelements.io/oauth` .

Some API provider require scopes let you limit the authorization to a subset of the data. When users grant authorization, they are shown the scopes and acknowledge that they grant access to them. The user must allow access to all or to none. The scopes that you set up in the application should match the **OAuth Scope**.

## OAuth 2.0 Authentication JSON

The JSON used to authenticate a connector instance looks like this:

```

{
  "element": {
    "key": ""
  },
  "providerData": {
    "code": ""
  },
  "configuration": {
    "oauth.callback.url": "",
    "oauth.api.key": "",
    "oauth.api.secret": "",
    "oauth.scope": ""
  },
  "tags": [
    ""
  ],
  "name": ""
}

```

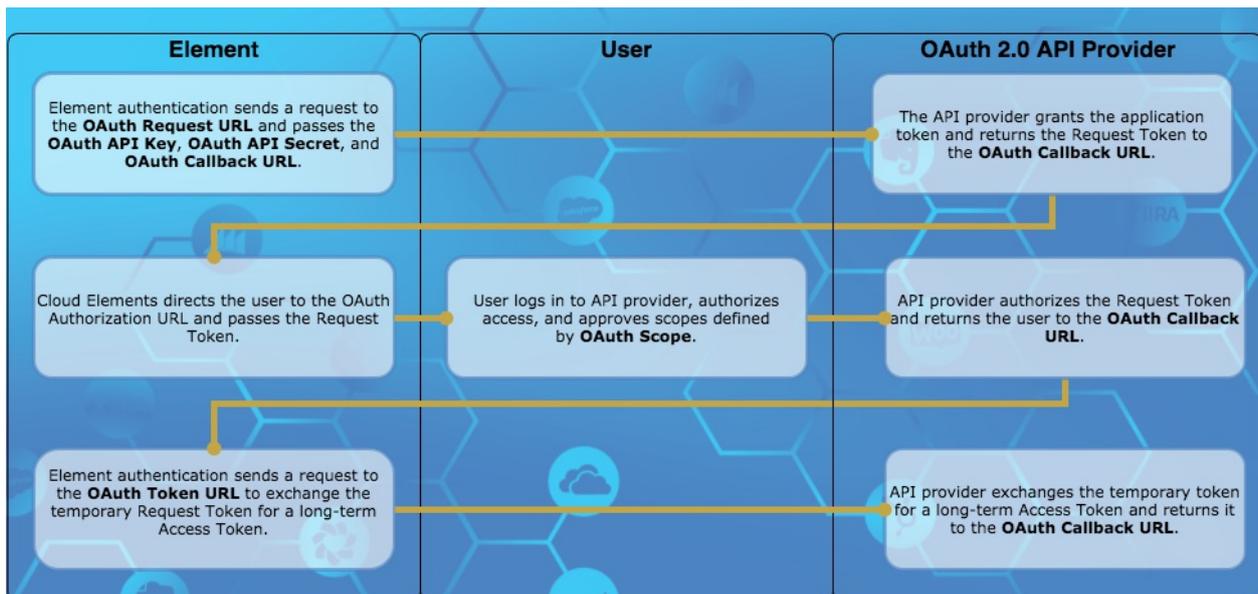
## OAuth 1.0

OAuth 1.0 is an older version of the OAuth protocol. Like OAuth 2.0, the OAuth 1.0 protocol lets external applications — your application or SAP — request authorization to access and update a users data without asking users for sensitive user names and passwords. OAuth 1.0 is also flexible with multiple different ways to implement it.

This section describes how SAP supports the [OAuth Core 1.0 Revision A](#) authentication flow. The typical OAuth 1.0 authorization flow is a three-legged authentication process:

1. Request temporary credentials. SAP requests temporary credentials in the form of a Request Token from the API provider.
2. Authorize. The user logs in to the API provider and grants the temporary Request Token access.
3. Token exchange. SAP exchanges the temporary Request Token for a more long-term access token.

### SAP OAuth 1.0 Flow



The typical OAuth 1.0 flow as supported by SAP includes the following steps:

1. SAP sends a request to the API provider's **OAuth Request URL** for a Request Token. SAP includes the following

parameters as part of the request:

- **OAuth API Key**
  - **OAuth API Secret**
  - **OAuth Callback URL**
2. The API provider returns the temporary Request Token and a secret associated with the token.
  3. SAP directs the user to the **OAuth Authorization URL** where they log in and grant access. If the request included **OAuth Scope**, the user must grant access to all scopes or none.
  4. The API provider authorizes the temporary Request Token and redirects the user back to the **OAuth Callback URL** with the authorized Request Token and an OAuth verifier.
  5. SAP sends a request to the **OAuth Token URL** and exchanges the authorized Request Token for a long-term Access Token.
  6. SAP associates a connector instance token with the Access Token. You will use that connector instance token in future requests.

## OAuth 1.0 Prerequisites

Before you can authenticate a connector instance, you must register your application with the API provider. Each registration is assigned some form of an API key (client id, client key, etc.) and secret. In SAP, these are the **OAuth API Key** and **OAuth API Secret**. You also usually need to provide a redirect URI which should match what you include in the **OAuth Callback URL**. If you want users to authenticate through SAP as opposed to programmatically through APIs in your application, use <https://auth.cloudelements.io/oauth>.

## OAuth 1.0 Authentication JSON

The JSON used to authenticate a connector instance looks like:

```
{
  "element": {
    "key": "twitter"
  },
  "providerData": {
    "oauth_token": "",
    "oauth_verifier": "",
    "secret": ""
  },
  "configuration": {
    "oauth.api.key": "",
    "oauth.api.secret": "",
    "oauth.callback.url": ""
  },
  "tags": [
    ""
  ],
  "name": ""
}
```

## Override Standard OAuth 2.0 Flow

Many API providers implement OAuth 2.0 in unique ways. When you create a connector and need to deviate from the standard SAP [OAuth 2.0 Flow](#), you can use specific endpoints to override each step in the flow. In each endpoint, use a combination of configurations, parameters, and hooks to override the standard OAuth 2.0 flow.

The resources available to override OAuth 2.0 flows are:

- [GET/oauth-authorize](#) — Overrides the first step in the OAuth 2.0 flow where SAP requests authorization on behalf of your app by redirecting a user to the API provider's Authorization URL.

- `POST/oauth-token-exchange` — Overrides the second step in the flow where SAP exchanges the code returned from the API provider for an access token.
- `POST/oauth-token-refresh` — Overrides the refresh step where SAP exchanges a refresh token for an updated access token.

Overriding the OAuth 2.0 flow is a complex task and the number of ways to perform the overrides are limited only to your imagination and ability to write JavaScript code. The steps below do not provide details of how to override specific steps, but serve as a guide to get you started. See [Custom Resources](#) for general steps to add a resource to a connector.

To add the `GET/oauth-authorize` resource:

1. Click **Add a new resource** at the top of the page.
2. In **SAP Resource Name** enter `/oauth-authorize`.
3. Do not enter anything in **Vendor Resource Name**.
4. Select only the **GET** method.

Your new resource should look like:

The screenshot shows the configuration form for a new resource. At the top, there are two input fields: "Open Connectors Resource Name" containing "/oauth-authorize" and "Vendor Resource Name" which is empty. A blue "Maps to →" button is positioned between them. Below these fields, there are three input fields for "Primary Key:", "Created Date Key:", and "Updated Date Key:", all of which are empty. To the right of these fields is a group of radio buttons for selecting HTTP methods: GET (checked), GET by ID, POST, PATCH, PUT, and DELETE. At the bottom right of the form, there are two buttons: "Go" and "Cancel".

5. Click **Go**.
6. Find the endpoint, and then click .
7. Change the description to something like: "Overriding the OAuth 2.0 authorization".
8. From **Resource Type**, select **OAuth ON AUTHORIZE URL**.
9. Some common configurations that you can use in the parameters of the endpoint or in the hooks to manipulate the request include:
  - OAuth API Key ( `oauth.api.key` )
  - OAuth API Secret ( `oauth.api.secret` )
  - OAuth Callback URL ( `oauth.callback.url` )
  - OAuth Authorization URL ( `oauth.authorization.url` )
  - OAuth Scope ( `oauth.scope` )

To add the `POST/oauth-token-exchange` resource:

1. Click **Add a new resource** at the top of the page.
2. In **SAP Resource Name** enter `/oauth-token-exchange`.
3. Do not enter anything in **Vendor Resource Name**.
4. Select only the **POST** method.

Your new resource should look like:

Open Connectors Resource Name  Maps to → Vendor Resource Name

Primary Key:

Created Date Key:

Updated Date Key:

GET  GET by ID  POST  PATCH  PUT  DELETE

5. Click **Go**.
6. Find the endpoint, and then click .
7. Change the description to something like: "Overriding the OAuth 2.0 token exchange".
8. From **Resource Type**, select **OAUTH ON TOKEN EXCHANGE**.
9. Some common configurations that you can use in the parameters of the endpoint or in the hooks to manipulate the request and response include:
  - o OAuth Callback URL ( `oauth.callback.url` )
  - o `oauth.user.refresh_token` (from the response)
  - o `oauth.user.token` (from the response)
  - o OAuth Refresh Interval (s) ( `oauth.user.refresh_interval` )
  - o The authorization code from the response

To add the `POST/oauth-token-refresh` resource:

1. Click **Add a new resource** at the top of the page.
2. In SAP **Resource Name** enter `/oauth-token-refresh`.
3. Do not enter anything in **Vendor Resource Name**.
4. Select only the **POST** method.

Your new resource should like:

Open Connectors Resource Name  Maps to → Vendor Resource Name

Primary Key:

Created Date Key:

Updated Date Key:

GET  GET by ID  POST  PATCH  PUT  DELETE

5. Click **Go**.
6. Find the endpoint, and then click .
7. Change the description to something like: "Overriding the OAuth 2.0 token refresh".
8. From **Resource Type**, select **OAUTH ON TOKEN REFRESH**.
9. Some common configurations that you can use in the parameters of the endpoint or in the hooks to manipulate the request and response include:
  - o OAuth Token Refresh URL ( `oauth.token.refresh_url` )
  - o OAuth Revoke Token URL ( `oauth.token.revoke_url` )
  - o OAuth Refresh Interval (s) ( `oauth.user.refresh_interval` )
  - o OAuth Token URL ( `oauth.token.url` )
  - o The refresh token code from the response
  - o `oauth.user.token` (from the response)