Cisco Webex Teams Authenticate a Connector

You can authenticate with Cisco Webex Teams to create your own instance of the Cisco Webex Teams connector through the UI or through APIs. Once authenticated, you can use the connector instance to access the different functionality offered by the Cisco Webex Teams platform.

Authenticate Through the UI

Use the UI to authenticate with Cisco Webex Teams and create a connector instance. Because you authenticate with Cisco Webex Teams via OAuth 2.0, all you need to do is add a name for the instance. After you create the instance, you'll log in to Cisco Webex Teams to authorize SAP Cloud Platform Open Connectors to access your account. For more information about authenticating a connector instance, see Authenticate a Connector Instance (UI).

After successfully authenticating, we give you several options for next steps. Make requests using the API docs associated with the instance, map the instance to a common resource, or use it in a formula template.

Authenticate Through API

SAP Cloud Platform Open Connectors recommends visiting the Cisco Webex Teams API Documentation and reviewing the information posted regarding the OAuth Scope and Permissions.

Step 1. Get Connectors OAuth Information

- HTTP Header: None
- HTTP Verb: GET
- Request URL: /elements/{keyOrId}/oauth/url
- Request Body: None

- Query Parameters:

- **apiKey**– the key obtained from registering your app with the provider

- **apiSecret** – the secret obtained from registering your app with the provider

- **callbackUrl** – the URL that you supplied to the provider when registering your app, state – any custom value that you want passed to the callback handler listening at the provided callback URL.

The result of this API invocation is an OAuth redirect URL from the endpoint. Your application should now redirect to this URL, which in turn will present the OAuth authentication and authorization page to the user. When the provided callback URL is executed, a code value will be returned, which is required for the Create Instance API.

Example cURL Command:

```
curl -X GET
-H 'Content-Type: application/json'
'https://api.openconnectors.us2.ext.hana.ondemand.com/elements/api-v2/eleme
nts/ciscospark/oauth/url?apiKey=insert_client_id&apiSecret=insert_client_se
cret&callbackUrl=https%3A%2F%2Fmycoolapp.com%2Fauth'
```

Response:

```
{
  "oauthUrl": "https://api.ciscospark.com/v1/authorize?scope=spark%3Arooms_
read+spark%3Arooms_write+spark%3Apeople_read+spark%3Amessages_read+spark%3A
messages_write+spark%3Amemberships_read+spark%3Amemberships_write+spark%3At
eams_write+spark%3Ateams_read+spark%3Ateam_memberships_read+spark%3Ateam_me
mberships_write+spark-admin%3Aorganizations_read++spark-admin%3Alicenses_re
ad+spark-admin%3Aroles_read+spark-admin%3Apeople_write+spark-admin%3Apeople
_read&response_type=code&redirect_uri=https%3A%2F%2Fauth.cloudelements.io%2
Foauth&state=ciscospark&client_id=C7219dadf8ebe5098eb054a1f2b98b7b73cfe296a
418b2926f2594e4664c08e37",
  "element": "ciscospark"
}
```

Upon successful authentication and authorization by the user, the endpoint will redirect to the callback URL you provided when you setup your application with the endpoint, in our example, https://www.mycoolapp.com/auth. The endpoint will also provide two query string parameters: "state" and "code". The value for the "state" parameter will be the name of the endpoint, e.g., `ciscospark` in our example, and the value for the "code" parameter is the code required by SAP Cloud Platform Open Connectors to retrieve the OAuth access and refresh tokens from the endpoint. If the user denies authentication and/or authorization, there will be a query string parameter called "error" instead of the "code" parameter. In this case, your application can handle the error gracefully.

Step 2. Create an Instance

To provision your Cisco Webex Teams Connector, use the `/instances` API.

Below is an example of the provisioning API call.

- **HTTP Headers**: Authorization- User , Organization
- **HTTP Verb**: POST
- **Request URL**: /instances
- **Request Body**: Required – see below
- **Query Parameters**: none

token is returned upon successful execution of this API. This token needs to be retained by the application for all subsequent requests involving this connector instance.

A sample request illustrating the /instances API is shown below.

HTTP Headers:

```
Authorization: User , Organization
```

This `instance.json` file must be included with your instance request. Please fill your information to provision. The "key" into SAP Cloud Platform Open Connectors Cisco Webex Teams is `ciscospark`. This will need to be entered in the `key` field below depending on which Connector you wish to instantiate.

```json
{
  "element": {
    "key": "ciscospark"
  },
  "configuration": {
    "oauth.api.secret": "xxxxxxxxxxxxxxxxx",
    "filter.response.nulls": "true",
    "oauth.callback.url": "https://auth.cloudelements.io/oauth",
    "oauth.scope": "spark:rooms_read spark:rooms_write spark:people_read sp
ark:messages_read spark:messages_write spark:memberships_read spark:members
hips_write spark:teams_write spark:teams_read spark:team_memberships_read s
park:team_memberships_write spark-admin:organizations_read  spark-admin:lic
enses_read spark-admin:roles_read spark-admin:people_write spark-admin:peop
le_read",
    "oauth.api.key": "xxxxxxxxxxxxxxxxxxxxxx",
    "event.notification.enabled": false,
    "oauth.user.refresh_time": null,
    "oauth.user.refresh_token": null
  },
  "name": "",
  "eventsEnabled": null,
  "providerData": {
    "code": "xxxxxxxxxxxxxxx",
    "state": "ciscospark",
    "debug": false,
    "secret": "xxxxxxxxxxxxxxxx"
  }
}
```

Here is an example cURL command to create an instance using /instances API.

Example Request:

```
curl -X POST
-H 'Authorization: User , Organization '
-H 'Content-Type: application/json'
-d @instance.json
'https://api.openconnectors.us2.ext.hana.ondemand.com/elements/api-v2/insta
nces'
```

If the user does not specify a required config entry, an error will result notifying her of which entries she is missing.

Below is a successful JSON response:

```json
{
  "id": 8728,
  "name": "Cisco Webex Teams",
  "hookName": "CiscoSpark",
  "key": "ciscospark",
  "description": "Add an Cisco Webex Teams Instance to connect your existing Cisco account to the Collaboration Hub, allowing you to manage channels, users, etc. across multiple Collaboration Elements. You will need your Cisco Webex Teams account information to add an instance.",
  "image": "elements/provider_ciscospark.png",
  "logo": "ciscowebexteams",
  "active": true,
  "deleted": false,
  "typeOauth": false,
  "trialAccount": false,
  "configuration": [
    {
      "id": 63207,
      "name": "OAuth API Secret",
      "key": "oauth.api.secret",
      "description": "Cisco Webex Teams OAuth API Secret",
      "resellerConfig": false,
      "companyConfig": false,
      "active": true,
      "internal": false,
      "groupControl": false,
      "displayOrder": 1,
      "type": "PASSWORD",
      "hideFromConsole": true,
      "required": true
    }...
  ],
  "resources": [...],
  "objects": [...],
  "models": [...],
  "transformationsEnabled": true,
  "bulkDownloadEnabled": false,
  "bulkUploadEnabled": false,
  "cloneable": true,
  "extendable": true,
  "beta": false,
  "authentication": {...},
  "hooks": [...],
  "extended": false,
  "useModelsForMetadata": true,
  "hub": "collaboration",
  "protocolType": "http",
  "parameters": [...],
  "private": false
}
```

Instance Configuration

The content in the `configuration` section or nested object in the body posted to the `POST /instances` or `PUT /instances/{id}` APIs varies depending on which connector is being instantiated. However, some configuration properties are common to all connectors and available to be configured for all connectors. These properties are -

- `event.notification.enabled` : This property is a `boolean` property, and determines if event reception (via `webhook` or `polling`) is enabled for the connector instance. This property defaults to *false*.
- `event.vendor.type` : When `event.notification.enabled` property is set to *true*, this property determines the mechanism to use to receive or fetch changed events from the service endpoint. The supported values are `webhook` and `polling` . Most connectors support one mechanism or the other, but some like Salesforce.com support both mechanisms. This property is *optional*.
- `event.notification.type` : This property can be used to determine how an event notification should be sent to the consumer of the connector instance, in most cases your application. Currently, `webhook` is the only supported value for this property. This means that when an event is received by the connector instance, it will get forwarded to the provided `event.notification.callback.url` via a `webhook` to you. This property is *optional*.
- `event.notification.callback.url` : As mentioned above, the value of this property is an `http` or `https` URL to which we will post the event for consumption by your application. This property is *optional*.
- `filter.response.nulls` : This property defaults to *true*, i.e., it's `boolean` property, and determines if `null` values in the response `JSON` should or should not be filtered from the response returned to the consuming application. By default, all `null` values are filtered from the response before sending the response to the consuming application.