SFTP Authenticate a Connector

Last Modified on 03/16/2020 10:06 pm EDT

You can authenticate with SFTP to create your own instance of the SFTP connector directly via hostname and port number. Once authenticated, you can use the connector instance to access the different functionality offered by the SFTP platform. For more information about generating private keys, see our documentation.

Create Instance Directly via Hostname and Port Number

The following is required to create a SFTP connector Instance:

- Hostname: e.g. `cm.mycoolapp.com`
- Port e.g. `22`
- SFTP Username
- SFTP Password
- Home Directory e.g. `/` or `/home`

Step 1. Create an Instance

To provision your SFTP 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

> ℹ **Note: Query Parameters**
> **Private Key** and **Private Key Password** - If this option is used, it can only take an RSA key as a private key, that is also base64 encoded. You need to take the whole key and base64 encode it and then paste that in the Instance during creation.
> For example, say the RSA key looks like this:
> ----BEGIN RSA PRIVATE KEY---- Proc-Type:4,ENCRYPTED
> DEK-Info: AES-128-CBC,31xxxxxxxxxxC0D412897

3SIrHX/gWKvUyqi3ZCjc1/zMVo+X7Q1fEGNOe20ftYjwYDJDMq3m1QPPjIVZI/kP
GyPrLMo9y315YaYbCtRodU4BaYKqRkWfK8y4+3Y37OfXX6W/O/55YLzp4A9FP6Ka
R0ulT3Cg8XIxxxxxxxxxxx7iRD/BqVDuaEcydq0wArp0FajnTy8
uzBKeflKb2BZvDyjuPBJELy8Stftzkea9MJiP4L0e9EreerTDuNNkJFP37SN9IyO
798pn1NmNAPaO4x62RvdhfG4s8Dgv+3kgkfK7lmdn9pSVB2WbpbxrrIXJOxmAI2y
Jos0HKh2uyZYllGhhfOTLZdedw7WBldfPCKyt6mjcwa4jVspt4uPh7P/Kd/xACRi
xxxxxxxxxxxxxxxxxxx
7CD1VPJPEqP5FsR4gdTdISJqI7rHIm15Mx0wZb+jqGy8UKvw9UxH47WiYGLjL3eN
zTMSoFbbfF+hadUspTvVwUmIUJ5y9cbcnKlLS3++8K6vKhrYAYPkhpqksWtLEvK8
jRbISL/h+f2BJDkKuL4+P0CNZtP1DFczmk8mg9NtvDkQpUN8Y/InfvME15ch4eGE
4Gp/uKl/IyAPfxIaxxxxxxxxxxOKqlp0m3X+
8wLbgpzd6fTnnSmqCQtHNdzjSEWyJTyb54hWQwwy+YZVZpVnQp0h3DC5Tj2I4UVb
CNuvP9pziCG6R2eowaZie9qrgPYgkIgsj/5cPYcp6i9XE/bEgqO8Vi6TIyjZJzWx
----END RSA PRIVATE KEY----

> You need to base64 encode the entire RSA key (between the highlighted text) and then provide it in the instance creation page.

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 SFTP is "sftp". This will need to be entered in the “key” field below depending on which connector you wish to instantiate.

```
{
  "element": {
    "key": "sftp"
  },
  "configuration" : {
      "hostname": "",
      "port": "",
      "username": "",
      "password":  "",
      "home_directory": ""
  },
  "tags": [
    ""
  ],
  "name": ""
}
```

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/instances'
```

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:

```
{
  "id": 1234,
  "name": "Test",
  "token": "VAnlQ/V28PT+M62kdajlsd90eHHtUJai+Efq8=",
  "element": {
    "id": 2753,
    "name": "SFTP connector Beta",
    "key": "sftp",
    "description": "Add a sftp Instance to connect your existing sftp account to the Documents Hub, allowing you to manage all of your Document activities across multiple Documents connectors. You will need your sftp account information to add an instance.",
    "image": "elements/provider_sftp.png",
    "active": true,
    "deleted": false,
    "typeOauth": false,
    "trialAccount": false,
    "configDescription": "Sftp",
    "transformationsEnabled": false,
    "bulkDownloadEnabled": false,
    "bulkUploadEnabled": false,
    "cloneable": false,
    "authentication": {
      "type": "basic"
    },
    "hub": "documents",
    "protocolType": "sftp",
    "parameters": [],
    "private": false
  },
  "provisionInteractions": [],
  "valid": true,
  "disabled": false,
  "maxCacheSize": 0,
  "cacheTimeToLive": 0,
  "configuration": {
    "event.notification.subscription.id": null,
    "password": "********",
    "hostname": "cm.mycoolapp.com",
    "event.poller.refresh_interval": "15",
    "event.notification.callback.url": null,
    "port": "22",
    "home_directory": "/home",
    "event.notification.signature.key": null,
    "event.vendor.type": "polling",
    "event.poller.configuration": "",
    "username": "USERNAME",
    "event.notification.enabled": "false"
  },
  "eventsEnabled": false,
  "traceLoggingEnabled": false,
  "cachingEnabled": false,
  "externalAuthentication": "none"
}
```

Note: Make sure you have straight quotes in your JSON files and cURL commands. Please use plain text formatting in your code. Make sure you do not have spaces after the in the cURL command.

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.