SAS Viya REST APIs

SAS Viya REST APIs, or simply Viya API, is nothing more than a REST API which is made available by SAS to the entire developing community and to data scientists who are already accustomed to working with open source tools.

In short, API (Application Programming Interface) is a system responsible for communication between a client and a server. REST (Representational State Transfer) is every API that follows the following standards, using the HTTP protocol:

1. Client-server
2. Stateless
3. Cacheable
4. Layered System
5. Code on demand (Optional)

Working with this set of APIs can give a lot of advantage when using SAS in some business process, since with just a few requests SAS Viya can be integrated into some web service.

For the developer and data science public, APIs are the foundation of several libraries capable of integrating languages such as Python, Lua, R, and Java, directly with SAS Viya.

That's why, if you're interested in understanding how this integration works, I invite you to continue reading this handy guide on how to consume viya APIs on any web service.

It is worth remembering that it is always good to consult the official documentation that is in the link: https://developer.sas.com/apis/rest/

Summary

Authentication and Access Tokens

Introduction

The authentication step is the first step you should take if you think about making use of the Viya SAS APIs. Every time you make a request (GET/POST/PUT/DELETE) to Viya, an access key is required for call validation. We call this access key `access_token`, and it is generated only once per session, and a duration can be set until it expires.

To generate an `access_token` you need to register a client on the Viya server. It within of this register that the `access_token` that will be used to validate every request made by our application will be generated.

This step may seem a bit complicated (it really is complicated) by having a large authentication and security system, but isn't this the price of having a robust security process? In addition, there are several tools that have already been developed to make our lives easier during authentication. I leave here the link of the blog written by Mary Kathryn Queen about a fantastic tool for authentication (Only for SAS Employees).

This process will be divided into 2 parts: Client Registration, and Obtaining an Access Token (via terminal and via implementation in a Python application).

It is essential to take into account 4 things:

1. Initially it is necessary to have administrator access on the server where SAS Viya is installed.
2. Have the curl installed inside the Viya SAS server.
3. Have an administrator user in the SAS Viya environment.
4. Have The Setting Cross-Origin Resource Sharing (CORS) option enabled (To enable this option you can follow the instructions on the website).

Registering a Client

The registration client is a unique process that we will do within the server to configure the client name, client password, and even the validity time of each `access_token`.

Consul Token

Go to your server and make sure you log in with a user of the sas group, or simply go as sudo:

```
sudo su
```

Go to the folder where the file containing the `consul token` is located

```
cd
/opt/sas/viya/config/etc/SASSecurityCertificateFramework/tokens/consul/de
fault
```

Copy the `consul token` and save it to use it later

```
cat client.token
```

Oauth token

`Oauth` is an authentication standard that protects our credentials when sending information using HTTP protocol. It is for the purpose of security that SAS provides an `Oauth token` to allow us to securely register our client. To get it, we need to provide two things:

1. A name that will later be assigned to the registered client. If you are creating an application for Open Banking that name is OpenBankingApp, but for educational purposes we will use the app name.

2. The `consul token` obtained in the previous step.

```
curl -X POST
"http://<hostname>/SASLogon/oauth/clients/consul?callback=false&serviceId
=<client-name>" \
      -H "X-Consul-Token: <value-of-consul-token-goes-here>"
```

Note: Also remember to replace `<hostname>` with the hostname of your SAS Viya server.

The return of this call will be a JSON string, in which we are only interested in the value of the field `access_token`, which is not yet our final `access_token`.

Copy and save it to use in the next step.

Registering a Client

In the previous step we asked the system to reserve our client name. In this step we will make another call providing this same name, and adding the client password (you can use 'Orion123'), and the `Oauth token` obtained in the previous step.

```
curl -X POST "http://<hostname>/SASLogon/oauth/clients" \
      -H "Content-Type: application/json" \
      -H "Authorization: Bearer <oauth-token>" \
      -d '{
      "client_id": "<client-name>",
      "client_secret": "<client-password>",
      "scope": ["openid"],
      "authorized_grant_types": ["password"],
      "access_token_validity": 43199
      }'
```

Note 1: By default, the validity time of the `access token` is set to 43199 seconds (12 hours), but this can be set by changing the value in the `"access_token_validity"` field.

Note 2: The name and password of the client are very important when making the calls of the APIs, so it is necessary that you do not forget them.

If the request was successfully made, we will receive a message similar to the following:

```
{"scope":["openid"],"client_id":"<client-
name>","resource_ids":["none"],"authorized_grant_types":["password","refr
esh_token"],
"access_token_validity":43199,"authorities":["uaa.none"],"lastModified":1
521124986406}
```

Ready! The application client has already been registered. We can now request an `access token` every time the application wishes to make requests.

Getting an Access Token

Now that there is already a registered client we can request an `access token` so that the application can call the Viya APIs.

You will be shown 2 ways to do this, first using curl from the terminal, and second, implementing a Python function so that the application can do this automatically.

Using CURL

Here we need to provide 4 important information:

1. Viya User
2. Viya user password
3. Client name
4. Client password

```
curl -X POST "http://<hostname>/SASLogon/oauth/token" \
      -H "Content-Type: application/x-www-form-urlencoded" \
      -d "grant_type=password&username=<user-viya>&password=<password-
```

```
viya>" \
      -u "<client-name>:<client-password>"
```

This call will return a JSON string, in which we are only interested in the value of the `'access_token'` field, which corresponds to the `access token` we were looking for.

Using Python

We've already seen how to register a client and how to get an `access token`, but it would be much more interesting to be able to automatically request tokens within our application. In this case we will use an application made in Python.

In the `obtainAccessTokenPythonUser()` function defined below, we need to pass 4 meters, which are those same 4 parameters that we use to get the token using the curl: user and password of our Viya environment, and name and password of the registered application.

```
import requests

appName = '<client-name>'
appPass = '<client-password>'
username = '<user-viya>'
password = '<paswword-viya>'
host = '<hostname>'

def obtainAccessTokenPythonUser(appName, appPass, username, password):
    url = "https://"+ host +"/SASLogon/oauth/token"

    headers = {"Content-Type":"application/x-www-form-urlencoded"}

    data = "grant_type=password&username=" + str(username)+ "&password="
+ str(password)

    auth = (str(appName), str(appPass))

    r = requests.post(url,headers=headers, data=data, timeout=30, auth =
auth, verify=False)
    print(r)
    print(r.json())
    if r.status_code == 200:
      return r.json()['access_token']
```

Note: You must have the library `requests` installed in your Python 3 environment.

If the call status is OK (200), the string `access token` is returned. This return can be stored in a variable to be used every time a call is made in the SAS Viya REST APIs.

References

You can find more information about Authentication and Client Registration at the following links:

- https://developer.sas.com/reference/auth/
- https://developer.sas.com/apis/rest/Topics/?shell#authentication-and-access-tokens
- https://blogs.sas.com/content/sgf/2019/01/25/authentication-to-sas-viya/
- https://communities.sas.com/t5/SAS-Communities-Library/Getting-Python-Talking-with-The-SAS-Viya-REST-API-Registering-a/ta-p/557882
- https://communities.sas.com/t5/SAS-Communities-Library/Getting-Python-Talking-with-The-SAS-Viya-REST-API-PART-2/ta-p/557933