Hewlett Packard Enterprise HPE Support Center

Issues accessing object store data from EZUA notebooks

Issue

When attempting to access S3 buckets from EZUA, customers are getting the below errors:

```
ClientError: Failed to connect to Amazon S3: An error occurred (500) when calling the ListBuckets operation (reached r
Failed to connect to Amazon S3: An error occurred (503) when calling the ListBuckets operation (reached max retries: 4
ClientError: An error occurred (500) when calling the ListBuckets operation (reached max retries: 4): Internal Server
```

Environment

EZUA 1.3

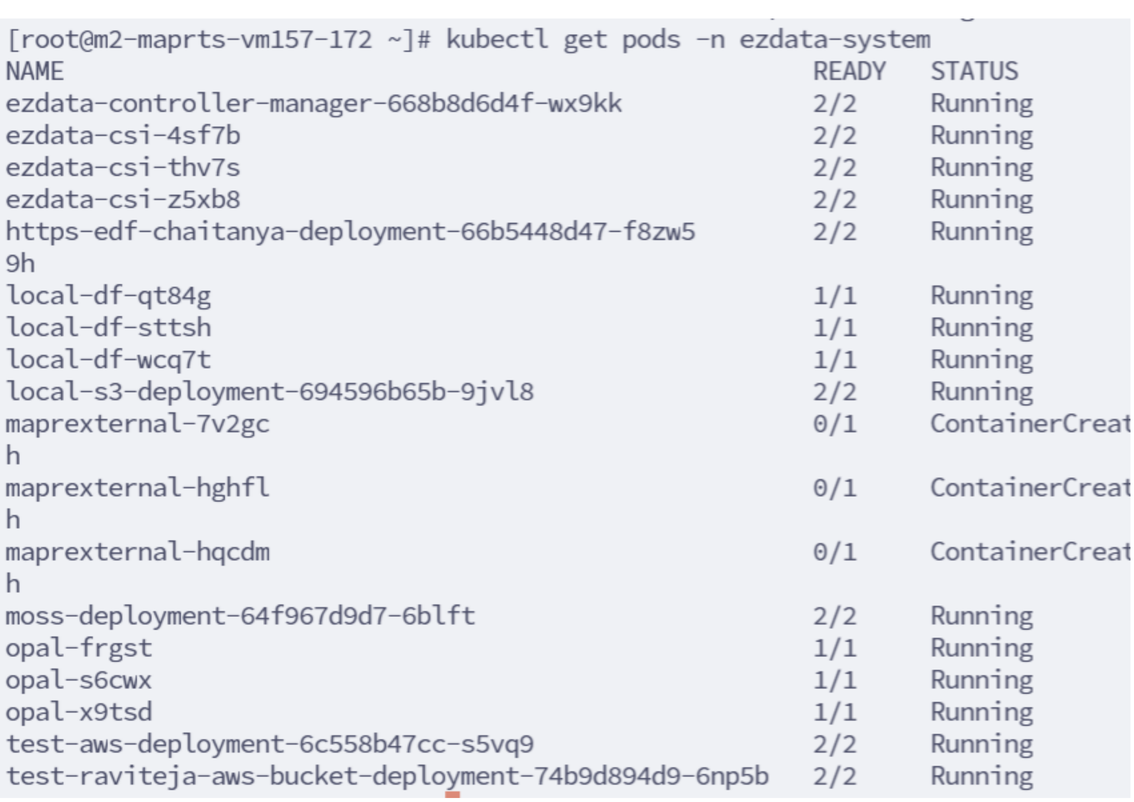
Cause

The above issue is primarily caused by giving incorrect authentication details such as specifying the internal endpoint and the access keys and secret keys of the external endpoint. From inside the EZUA platform, if we are trying to access any s3 bucket which is added as object store data, then authentication may fail in this situation.

Resolution

When you add an external data source into the EZUA platform, such as an AWS S3 bucket or an Ezmeral Data Fabric S3 bucket, immediately a new pod will be created similar to the name given for the datasource inside the "ezdata-system" namespace. We can verify this using the below command:

```
# kubectl get pods -n ezdata-system
```



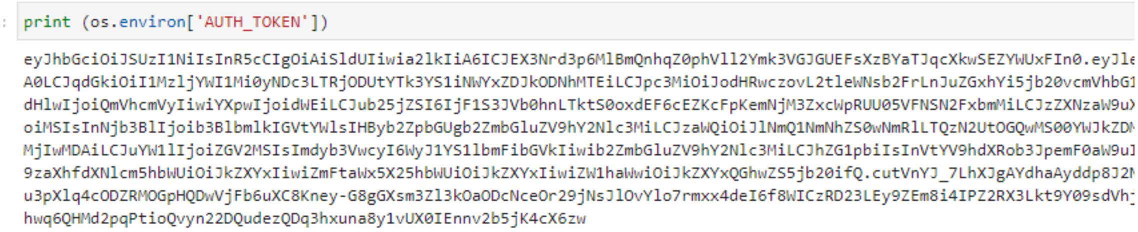
```
[root@m2-maprts-vm157-172 ~]# kubectl get pods -n ezdata-system
NAME                                                   READY   STATUS
ezdata-controller-manager-668b8d6d4f-wx9kk             2/2     Running
ezdata-csi-4sf7b                                       2/2     Running
ezdata-csi-thv7s                                       2/2     Running
ezdata-csi-z5xb8                                       2/2     Running
https-edf-chaitanya-deployment-66b5448d47-f8zw5        2/2     Running
9h
local-df-qt84g                                         1/1     Running
local-df-sttsh                                         1/1     Running
local-df-wcq7t                                         1/1     Running
local-s3-deployment-694596b65b-9jvl8                   2/2     Running
maprexternal-7v2gc                                     0/1     ContainerCreat
h
maprexternal-hghfl                                     0/1     ContainerCreat
h
maprexternal-hqcdm                                     0/1     ContainerCreat
h
moss-deployment-64f967d9d7-6blft                       2/2     Running
opal-frgst                                             1/1     Running
opal-s6cwx                                             1/1     Running
opal-x9tsd                                             1/1     Running
test-aws-deployment-6c558b47cc-s5vq9                   2/2     Running
test-raviteja-aws-bucket-deployment-74b9d894d9-6np5b   2/2     Running
```

We will get the list of all available pods, and from the above screenshot we see "test-raviteja-aws-bucket-deployment-74b9d894d9-6np5b" is the newly created pod where the data source is added into EZUA.

Whenever we get S3 authentication errors, either when trying to connect to them using notebooks or spark jobs, then firstly we can check the logs of the above pod to see if we have any errors which are causing the authentication failure.

If everything seems normal in the logs, and if we are trying to access the s3 bucket using an internal endpoint which is generated after adding the data source in our EZUA platform (via http://test-raviteja-aws-bucket-service.ezdata-system.svc.cluster.local:30000), then we do not need to send any access keys or secret keys along with the above endpoint to access it since the authentication gets its data from a variable named **AUTH_TOKEN,** which is generated by default and added in our **jupyterhub notebooks.**

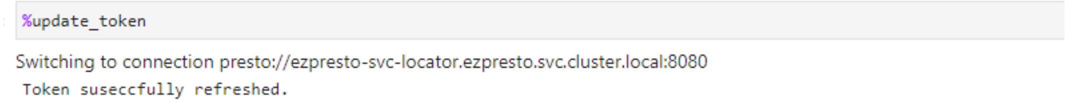
You can also fetch this information using the below command:



```
print (os.environ['AUTH_TOKEN'])
```

```
eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6ICJEX3Nrd3p6MlBmQnhqZ0phVll2Ymk3VGJGUEFsXzBYaTJqcXkwSEZYWUxFIn0.eyJle
A0LCJqdGkiOiI1MzljYWI1Mi0yNDc3LTRjODUtYTk3YS1iNWYxZDJkODNhMTEiLCJpc3MiOiJodHRwczovL2tleWNsb2FrLnJuZGxhYi5jb20vcmVhbG1
dHlwIjoiQmVhcmVyIiwiYXpwIjoidWEiLCJub25jZSI6IjF1S3JVb0hnLTktS0oxdEF6cEZKcFpKemNjM3ZxcWpRUU05VFNSN2FxbmMiLCJzZXNzaW9uX
oiMSIsInNjb3BlIjoib3BlbmlkIGVtYWlsIHByb2ZpbGUgb2ZmbGluZV9hY2Nlc3MiLCJzaWQiOiJlNmQ1NmNhZS0wNmRlLTQzN2UtOGQwMS00YWJkZD
MjIwMDAiLCJuYW1lIjoiZGV2MSIsImdyb3VwcyI6WyJ1YS1lbmFibGVkIiwib2ZmbGluZV9hY2Nlc3MiLCJhZG1pbiIsInVtYV9hdXRob3JpemF0aW9uI
9zaXhfdXNlcm5hbWUiOiJkZXYxIiwiZmFtaWx5X25hbWUiOiJkZXYxIiwiZW1haWwiOiJkZXYxQGhwZS5jb20ifQ.cutVnYJ_7LhXJgAYdhaAyddp8J2
u3pXlq4cODZRMOGpHQDwVjFb6uXC8Kney-G8gGXsm3Zl3kOaODcNceOr29jNsJlOvYlo7rmxx4deI6f8WICzRD23LEy9ZEm8i4IPZ2RX3Lkt9Y09sdVh
hwq6QHMd2pqPtioQvyn22DQudezQDq3hxuna8y1vUX0IEnnv2b5jK4cX6zw
```

You can copy the above JSON Web Token (jwt) and check its validity at the jwt.io website. In rare cases we want to refresh a token and update it by doing the following:



```
%update_token
```

```
Switching to connection presto://ezpresto-svc-locator.ezpresto.svc.cluster.local:8080
Token suseccfully refreshed.
```

If after doing all of the above we still get the same error and we want to check further on the issue, we can use the pods in the "token-service" namespace which is responsible for generating the above JWT token and validating those in our EZUA.

```
# kubectl get pods -n token-service
```

Then we can check the token-service pod logs to debug the issue further. Mostly the above issue would be solved if we use boto3 client to use AUTH_TOKEN for authentication rather than external access key and secret key for auth.

An example function is added below:

```
import boto3
import os

# Define endpoint URL for your S3 service
os.environ["AWS_ENDPOINT_URL"] = "http://test-raviteja-aws-bucket-service.ezdata-system.svc.cluster.local:30000"

s3 = boto3.client("s3")

response = s3.list_buckets()
print(response)
```

The above code snippet prints all of the available buckets in our data source.