Amazon Web Services (AWS) S3 Uploading, Downloading and Syncing Locally, on EC2 and in Docker Container

Fan Wang

2020-10-15

Contents

1 S3 Usages

Go to the **RMD**, **PDF**, or **HTML** version of this file. Go back to Python Code Examples Repository (bookdown site) or the pyfan Package (API).

1.1 Upload Local File to S3

A program runs either locally or on a remote EC2 machine inside a docker container. Upon exit, data does not persist in the docker container and needs to be exported to be saved. The idea is to export program images, csv files, json files, etc to S3 when these are generated, if the program detects that it is been executed on an EC2 machine (in a container).

First, inside the program, detect platform status. For Docker Container on EC2, AWS Linux 2 has platform.release of something like *4.14.193-194.317.amzn2.x86_64*.

```
import platform as platform
print(platform.release())
# This assums using an EC2 instance where amzn is in platform name
```

```
## 10
```

```
if 'amzn' in platform.release():
    s3_status = True
else:
    s3_status = False
print(s3_status)
```

```
## False
```

Second, on s3, create a bucket, *fans3testbucket* for example (no underscore in name allowed). Before doing this, set up AWS Access Key ID and AWS Secrete Acccess KEy in */Users/fan/.aws* folder so that boto3 can access s3 from computer. Upon successful completion of the push, the file can be accessed at https://fans3testbucket.s3.amazonaws.com/_data/iris_s3.dta.

```
import boto3
s3 = boto3.client('s3')
spn_local_path_file_name = "C:/Users/fan/Py4Econ/aws/setup/_data/iris_s3.dta"
str_bucket_name = "fans3testbucket"
spn_remote_path_file_name = "_data/iris_s3.dta"
s3.upload_file(spn_local_path_file_name, str_bucket_name, spn_remote_path_file_name)
```

1.2 Download File from S3 to Local Machine

On a local computer, download a particular file from S3. Download back the file we just uploaded onto S3.

```
import boto3
s3 = boto3.client('s3')
spn_local_path_file_name = "C:/Users/fan/Py4Econ/aws/setup/_data/iris_s3_downloaded.dta"
str_bucket_name = "fans3testbucket"
spn_remote_path_file_name = "_data/iris_s3.dta"
s3.download_file(str_bucket_name, spn_remote_path_file_name, spn_local_path_file_name)
```

1.3 Download File from S3 to EC2 Machine

On a EC2 machine, *Amazon Linux 2 AMI*, for example. Install pip, and install boto3, then download file to the */data/* folder.

```
# ssh into EC2 linux 2 AMI
ssh -i "G:/repos/ThaiJMP/boto3aws/aws_ec2/pem/fan_wang-key-pair-us_east_nv.pem" ec2-user@3.81.101.142
# generate data folder
mkdir data
# install boto3
sudo yum install python-pip python3-wheel && Pip install boto3 --user
# try download file using boto3
# go into python
python
```

Now inside python, download the *iris_s3.dta* to the data folder under root in the EC2 Machine.

```
import boto3
s3 = boto3.client('s3')
spn_ec2_path_file_name = "/home/ec2-user/data/iris_s3_downloaded.dta"
str_bucket_name = "fans3testbucket"
spn_s3_path_file_name = "_data/iris_s3.dta"
s3.download_file(str_bucket_name, spn_s3_path_file_name, spn_ec2_path_file_name)
```

1.4 Download File from S3 to Active Docker Container

Working inside an active docker container.

First activate and enter into a docker container:

```
# inside EC2 AMI Linux 2, start dockers
sudo service docker start
sudo service docker status
# see docker images
docker images
# run docker container and enter inside
```

```
docker run -t -i fanconda /bin/bash
# make a data directory and a esti subdirectory
mkdir data
cd data
mkdir esti
# enter python
python
```

Inside docker, which has boto3 installed already. The Path is different than on EC2, the path root structure is shorter, but otherwise the same.

```
import boto3
s3 = boto3.client('s3')
spn_container_path_file_name = "/data/esti/iris_s3_downloaded.dta"
str_bucket_name = "fans3testbucket"
spn_s3_path_file_name = "_data/iris_s3.dta"
s3.download_file(str_bucket_name, spn_s3_path_file_name, spn_container_path_file_name)
```

1.5 Forward and Backward Slashes

Following up on the uploading example above, suppose that rather than using forward slash, backward slash is used, then AWS gets confused about folder. This will not appear under the _data folder, but will appear as a file with file name: *_data/iris_s3.dta* under the bucket.

Note that the *folder* for S3 is a GUI trick, but still, we want to use forward slash properly, so that all double backslash that might be generated by default path tools need to be converted to forward slashes

```
import os
# This generates a file directly under bucket _data\iris_s3:
spn_remote_path_file_name_backslash = "_data\\iris_s3_slashbackforward.dta"
s3.upload_file(spn_local_path_file_name, str_bucket_name, spn_remote_path_file_name_backslash)
# This allows the folder structure to be clickable:
spn_remote_path_file_name_forwardslash = spn_remote_path_file_name_backslash.replace(os.sep, '/')
s3.upload_file(spn_local_path_file_name, str_bucket_name, spn_remote_path_file_name_forwardslash)
# Print slashs
print(f'{spn_remote_path_file_name_backslash=}')
```

```
## spn_remote_path_file_name_backslash='_data\\iris_s3_slashbackforward.dta'
```

```
print(f'{spn_remote_path_file_name_forwardslash=}')
```

```
## spn_remote_path_file_name_forwardslash='_data/iris_s3_slashbackforward.dta'
```

1.6 Sync Local Drive with S3 of a Particular File Type

Boto3 does not offer directory upload/download for s3. Needs to rely on aws command line.

To sync local folder with all files from a particular AWS folder, exclude image files:

```
# CD into a directory
cd /d "G:\S3\fanconda202010\esti"
# Make a new directory making S3 Directory Name
mkdir e_20201025x_esr_medtst_list_tKap_mlt_ce1a2
# cd into the directory just made
cd /d "G:\S3\thaijmp202010\esti\e_20201025x_esr_medtst_list_tKap_mlt_ce1a2"
# copy all results from the s3 folder's subfolders including subfolders, excluding images
aws s3 cp ^
```

```
    s3://fanconda202010/esti/e_20201025x_esr_medtst_list_tKap_mlt_ce1a2/ . ^
    --recursive --exclude "*.png"
```