**Comprehensive Guide to Download Files From S3 with Python**

Using AWS SDK for Python can be confusing. First of all, there seems to be two different ones (Boto and Boto3). Even if you choose one, either one of them seems to have multiple ways to authenticate and connect to AWS services. Googling solutions can quickly become confusing as you may find different variations of code examples. If you are just getting into AWS, this can be scary. In this post, I will explain the different and give you the code examples that work by using the example of downloading files from S3.

Boto is the older version of Python AWS SDK. Boto3 is the newer version. Boto3 doesn’t mean it is for Python 3. It works for Python 2.6 or 2.7. All the code provided in this post will work for both Python versions. As Boto3 is rewritten from ground up, the code you write with boto3 is different from Boto. You can read further about the change made in Boto3 [here](http://boto3.readthedocs.io/en/latest/guide/migration.html).

Generally speaking, you should use Boto3 if you are writing new programs. It is the new and improved version of Boto. I personally find it more efficient and easier to use. You also need to know Boto to maintain legacy code.

Before you start, you need to install boto and boto3. All the code work for both Python 2.7 and 3. You can use either one of them

[cc lang="bash" tab\_size="4" lines="-1" theme="mac-classic" line\_numbers="false"]

Pip install boto

Pip install boto3

[/cc]

1. **Downloading S3 Files With Boto3**

Boto3 provides super-easy way to configure credentials and access to AWS resources. To connect to S3, you can either create a S3 resorce or S3 client. [Resource](http://boto3.readthedocs.io/en/latest/guide/resources.html) is an object-oriented interface to AWS and provides higher-level abstraction while [Client](http://boto3.readthedocs.io/en/latest/guide/clients.html) is a low-level interface to AWS whose methods map close to 1:1 with service APIs.

**Using Resource**

You can create s3 resources with boto3.resource(‘s3’). It will use the AWS credentials configured in your AWS CLI (see [here](https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-getting-started.html)). This is perfect when you want to use the code between different environments as the credentials come from environment variable and you do not need to hardcode it. Once you have the resources, create the bucket object and use the download\_file method. [This page](http://boto3.readthedocs.io/en/latest/guide/s3-example-download-file.html) shows different s3 methods that you can use with resource.

[cc lang="python" tab\_size="4" lines="-1" theme="GeSHi" line\_numbers="false"]

import boto3

import botocore

def download\_file\_with\_resource(bucket\_name, key, local\_path):

s3 = boto3.resource('s3')

s3.Bucket(bucket\_name).download\_file(key, local\_path)

print('Downloaded File with boto3 resource')

bucket\_name = '<your bucket name>'

key = '<folder…/filename>'

local\_path = '<e.g. ./log.txt>'

download\_file\_with\_resource(bucket\_name, key, local\_path)

[/cc]

Troubleshoot Tip: An error occurred (404) when calling the HeadObject operation: Not Found

If you still get this error after triple-checking bucket name and object key, make sure your key does not start with ‘/’. For example, key = ‘/2018-02-27/log.txt’ should be key = ‘2018-02-27/log.txt’ in the code above.

**Using Client**

With Client, you can specify the credentials. As opposed to Boto, you do not need to specify the region. Boto3 handles it for you. You can also configure multiple credentials in AWS CLI and choose it to connect to non-default S3 with client. This documentation [here](http://boto3.readthedocs.io/en/latest/guide/configuration.html) is for further details on configuring credentials. [This page](http://boto3.readthedocs.io/en/latest/reference/services/s3.html#client) shows the list of methods you can use with client. Note that download\_file method argument is different from the one used with resource.

[cc lang="python" tab\_size="4" lines="-1" theme="GeSHi" line\_numbers="false"]

import boto3

import botocore

def download\_file\_with\_client(access\_key, secret\_key, bucket\_name, key, local\_path):

client = boto3.client('s3',\

aws\_access\_key\_id=access\_key,\

aws\_secret\_access\_key=secret\_key,\

)

client.download\_file(bucket\_name, key, local\_path)

print('Downloaded frile with boto3 client')

access\_key = '<Access Key>'

secret\_key = '<Secret Key>'

bucket\_name = '<your bucket name>'

key = '<folder…/filename>'

local\_path = '<e.g. ./log.txt>'

download\_file\_with\_client(access\_key, secret\_key, bucket\_name, key, local\_path)

[/cc]

1. **Downloading S3 Files With Boto**

Boto requires more parameters compared to Boto3 and you need to know a little bit more about the package to make it work.

Here are some common errors and how to handle them.

**Error 1:**

sl.CertificateError: hostname <your bucket name>.s3.amazonaws.com' doesn't match either of '\*.s3.amazonaws.com', 's3.amazonaws.com'

You will get this error when your bucket contain ‘.’ Without specifying OrdinaryCallingFormat() in connect\_s3().

**Error 2:**

Cannot resolve boto.exception.S3ResponseError: S3ResponseError: 301 Moved Permanently

You will get this error without specifying the host with region if you are not using one of the default US region. Otherwise, it will use s3.amazonaws.com as a host name which assumes your bucket is created in the default US region. Alternatively, you can use boto.s3.connect\_to\_region() to specify the region.

[Here](http://docs.ceph.com/docs/giant/radosgw/s3/python/) is the further reference on Python S3 with Boto.

**Using boto.connect\_s3() and default credentials**

Let’s start with using the default credential configures with AWS CLI. This is the equivalent of the first code in Boto3. You can see how simplified Boto3 is.

[cc lang="python" tab\_size="4" lines="-1" theme="GeSHi" line\_numbers="false"]

import boto

import boto.s3.connection

def download\_data(region, bucket\_name, key, local\_path):

conn = boto.connect\_s3(

host='s3-{}.amazonaws.com'.format(region),\

calling\_format = boto.s3.connection.OrdinaryCallingFormat())

bucket = conn.get\_bucket(bucket\_name)

key = bucket.get\_key(key)

key.get\_contents\_to\_filename(local\_path)

print('Downloaded File {} to {}'.format(key, local\_path))

region = '<Region e.g. ap-southeast-2>'

bucket\_name = '<your bucket name>'

key = '<folder…/filename>'

local\_path = '<e.g. ./log.txt>'

download\_data(region, bucket\_name, key, local\_path)

[/cc]

**Using boto.connect\_s3() with custom credentials**

Of course you can configure the credential within the code.

[cc lang="python" tab\_size="4" lines="-1" theme="GeSHi" line\_numbers="false"]

import boto

import boto.s3.connection

def download\_data\_connect\_s3(access\_key, secret\_key, region, bucket\_name, key, local\_path):

conn = boto.connect\_s3(aws\_access\_key\_id = access\_key,\

aws\_secret\_access\_key = secret\_key,\

host='s3-{}.amazonaws.com'.format(region),\

calling\_format = boto.s3.connection.OrdinaryCallingFormat()\

)

bucket = conn.get\_bucket(bucket\_name)

key = bucket.get\_key(key)

key.get\_contents\_to\_filename(local\_path)

print('Downloaded File {} to {}'.format(key, local\_path))

region = '<Region e.g. ap-southeast-2>'

access\_key = '<Access Key>'

secret\_key = '<Secret Key>'

bucket\_name = '<your bucket name>'

key = '<folder…/filename>'

local\_path = '<e.g. ./log.txt>'

download\_data\_connect\_s3(access\_key, secret\_key, region, bucket\_name, key, local\_path)

[/cc]

**Using boto.s3.connect\_to\_region()**

The main difference from connect\_s3() is that this function takes region as an argument while the region had to be defined in the host url in connect\_s3(). This is slightly simpler implementation to specify the region.

[cc lang="python" tab\_size="4" lines="-1" theme="GeSHi" line\_numbers="false"]

import boto

import boto.s3.connection

def download\_data\_connect\_to\_region(region, access\_key, secret\_key, bucket\_name, key, local\_path):

'''This will use connect\_to\_region() function in boto'''

conn = boto.s3.connect\_to\_region(

region\_name=region,

aws\_access\_key\_id=access\_key,

aws\_secret\_access\_key=secret\_key,

calling\_format=boto.s3.connection.OrdinaryCallingFormat()

)

bucket = conn.get\_bucket(bucket\_name)

key = bucket.get\_key(key)

key.get\_contents\_to\_filename(local\_path)

print('Downloaded File {} to {}'.format(key, local\_path))

region = '<Region e.g. ap-southeast-2>'

access\_key = '<Access Key>'

secret\_key = '<Secret Key>'

bucket\_name = '<your bucket name>'

key = '<folder…/filename>'

local\_path = '<e.g. ./log.txt>'

download\_data\_connect\_to\_region(region, access\_key, secret\_key, bucket\_name, key, local\_path)

[/cc]

Hope this clarifies a few things!

If you are interested in moving data to S3 and Redshift, check out this post: [Data Engineering in S3 and Redshift with Python](https://www.mydatahack.com/moving-data-in-s3-and-redshift-with-python/).