**How to Deploy Python Lambda Function with Serverless in AWS**

To deploy a [AWS lambda function](https://docs.aws.amazon.com/lambda/latest/dg/welcome.html) written in Python, you need to package all the modules that are used in your code before pushing it to your environment. [Serverless](https://serverless.com/framework/docs/providers/aws/guide/intro/) is a great lambda deployment framework. It takes care of all the packaging and deployment.

With Python, the best approach to develop lambda function is to use Linux or Mac. I found the module installed and packaged from Windows machine sometimes doesn’t work once I pushed to AWS. Serverless needs Docker. Docker and Serverless do not work very well on Windows. I had so much trouble configuring my Windows machine to make them work together. So, my approach is to run a Linux machine in a virtual box from Windows and develop and deploy lambda functions from there. In this post, I am using Ubuntu 16.0.4 LTS.

Prior to this, you have to set up the environment: [**Setting up Docker and Serverless Development Environment in Ubuntu**](https://www.mydatahack.com/setting-up-docker-and-serverless-development-environment-in-ubuntu/).

This post is based on the serverless blog ([How to Handle your Python packaging in Lambda with Serverless plugins](https://serverless.com/blog/serverless-python-packaging/)) with a few additional information. The instruction provided there is great. But, it is missing a few minor details which may become difficult if you have no idea what serverless does. Well, let’s get started.

**Steps**

1. **Create and Activate Virtual Environment for Lambda Development**

Make sure that you have the latest Python 3. You can name your virtual environment whatever. Here, we are using the environment called lambda. You can change the path to lambda, depending on your installation. To install Python 3 and create virtual environments, see the post [here](https://www.mydatahack.com/how-to-install-python-3-and-create-virtual-environment-in-ubuntu/).

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

source /home/user/python3/lambda/bin/activate

[/cc]

1. **Install Required Packages**

In this exercise, we will use numpy. Let’s also install psycopg2 and import it in the code to see how it works, too. For Lambda function, you need to install the binary version of psycopg2. Once you activate the virtual environment, pip will install the packages into that environment.

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

pip install numpy

pip install psycopg2-binary

[/cc]

1. **Go to the folder where you create Serverless project.**

I created a tmp folder in my home path (/home/user/).

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

mkdir /home/user/tmp/

[/cc]

1. **Create Python Serverless Project**

Go to the tmp folder created above and create project. If you set the path argument to numpy-test as in the example, it will create the folder. If you already have the folder with the same name, it will give you an error.

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

cd /home/user/tmp/

serverless create \

--template aws-python3 \

--name numpy-test \

--path numpy-test

[/cc]

1. **Create Function**

After running serverless create command, you will see handler.py in the numpy-test folder. Let’s edit it as in the example [here](https://serverless.com/blog/serverless-python-packaging/) with additional psycopg2 import statement. Test the function to see if it works.

[cc lang="python" tab\_size="4" lines="-1"]

[/cc]

1. **Add serverless-python-requirements plugin.**

You need to install the node package as below.

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

npm init

npm install –save serverless-python-requirements

[/cc]

1. **Edit serverless.yml file**

Once the function is working, you need to edit the config file for deployment. The file, serverless.yml, is in the lambda-test folder. You need to edit functions and add plugin information.

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

service: numpy-test

provider:

name: aws

runtime: python3.6

functions:

numpy:

handler: handler.main

plugins:

- serverless-python-requirements

custom:

pythonRequirements:

dockerizePip: non-linux

[\cc]

1. **Deploy the function**

The most important thing about this step is to specify the region. Without the region, it will deploy the function to US East (N. Virginia). See the further information on serverless deploy command [here](https://serverless.com/framework/docs/providers/aws/guide/deploying/).

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

Serverless deploy --region ap-southeast-2

[/cc]

You can see the deployment is complete.

1. **Test the function from AWS Management Console**

Go to AWS Management Console to see if you have numpy-test function. By default, it is tagged as dev, which means the name of your function is numpy-test-dev-numpy.

Create an event with no parameter (and name it as numpy) and click Test.

You can see the function is successfully executed on the console.

Yay!

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

[/cc]