**LAMBDA Learnings**

**Why in the code boto3 module is not required?**

The code you provided is designed to handle CloudWatch Logs events, specifically those delivered with AWS Lambda. In this scenario, you receive the CloudWatch Logs event directly as a parameter (`event`) in your Lambda function

The Lambda service manages the integration with CloudWatch Logs, so you don't need to use Boto3 directly in your Lambda function to retrieve CloudWatch Logs. The CloudWatch Logs events are delivered automatically to your Lambda function when logs are written to the specified log group.

The primary purpose of Boto3 is to interact with AWS services, and in this case, AWS Lambda takes care of delivering the CloudWatch Logs events to your Lambda function. The code you provided focuses on processing the received CloudWatch Logs events, decompressing the data, and uploading it to MongoDB.

If, in the future, you need to interact with other AWS services or perform additional actions before processing CloudWatch Logs events, you might need to use Boto3 in your Lambda function. For the current use case, where you're processing CloudWatch Logs events directly, Boto3 is not necessary.

**Note**: The logs of a cloudwatch log grp is encoded and compressed in a format mentioned below and is received by cloudwatch logs events (in "test" button of lambda console)

**EVENT Type: (cloudwatch log event type)**

**{**

**"awslogs": {**

**"data": "H4sIAAAAAAAAAHWPwQqCQBCGX0Xm7EFtK+smZBEUgXoLCdMhFtKV3akI8d0bLYmibvPPN3wz00CJxmQnTO41whwWQRIctmEcB6sQbFC3CjW3XW8kxpOpP+OC22d1Wml1qZkQGtoMsScxaczKN3plG8zlaHIta5KqWsozoTYw3/djzwhpLwivWFGHGpAFe7DL68JlBUk+l7KSN7tCOEJ4M3/qOI49vMHj+zCKdlFqLaU2ZHV2a4Ct/an0/ivdX8oYc1UVX860fQDQiMdxRQEAAA=="**

**}**

**}**

All the logs are inside the **data** digest of **awslogs.** So, now in the lambda code you have to decode and decompress the digest to get the logs.

**Note:** If you get any errors like **pymongo module not found** …anything like that then you have to create a lambda layer in which the modules that you are using in your code you have to upload it as **.zip** and attach it to the lambda function. For such purposes **Lambda** layers are being used.

Link for this: [**https://www.linkedin.com/pulse/add-external-python-libraries-aws-lambda-using-layers-gabe-olokun/**](https://www.linkedin.com/pulse/add-external-python-libraries-aws-lambda-using-layers-gabe-olokun/)

**Note:** First deploy the code and then test if you wanna test it by clicking the test button.  
  
code: This code takes the logs from specific cloudwatch log grp and send it to MongoDB

import base64

import zlib

import json

from pymongo import MongoClient

import boto3

# MongoDB settings

mongo\_connection\_string = "mongodb+srv://admin:TeleglobalPune2001@cluster0.vpx0ozf.mongodb.net/?retryWrites=true&w=majority"

mongo\_database\_name = "test-db"

mongo\_collection\_name = "vpc\_flow\_logs"

# Hardcoded log group name

log\_group = "dr-vpc-flow-logs-cloudwatch-log-grp"

def lambda\_handler(event, context):

try:

if not log\_group:

print("Invalid or missing log group name.")

return

print(f"Received CloudWatch Logs event from log group: {log\_group}")

# Extract the base64-encoded and compressed log data

log\_data = event['awslogs']['data']

decoded\_data = base64.b64decode(log\_data)

decompressed\_data = zlib.decompress(decoded\_data, zlib.MAX\_WBITS | 16)

# Convert the decompressed data to a string

log\_events\_str = decompressed\_data.decode('utf-8')

log\_events = log\_events\_str.splitlines()

# Print the first log event

if log\_events:

print(f"First log event:\n{log\_events[0]}")

# Process and upload logs to MongoDB

upload\_logs\_to\_mongodb(log\_events)

except Exception as e:

print(f"An error occurred: {str(e)}")

def upload\_logs\_to\_mongodb(logs):

client = MongoClient(mongo\_connection\_string)

db = client[mongo\_database\_name]

collection = db[mongo\_collection\_name]

for log in logs:

try:

log\_dict = json.loads(log)

# Insert the log dictionary into MongoDB

collection.insert\_one(log\_dict)

except json.JSONDecodeError:

print(f"Invalid JSON log format: {log}")

* **Cloud watch trigger is setupd**
* **Layer used** pymongo module **layer**