Practical Worksheet 10

Version: 1.0 Date: 12/04/2018 Author: David Glance

## Learning Objectives

1. Write an application that communicates using MQTT
2. Register a device using AWS IoT
3. Publish and subscribe to topics
4. Create a lambda function to handle IoT events and update a DynamoDB table

## Technologies Covered

AWS IoT

AWS Lambda

AWS DynamoDB

Python

Boto

## IoT Device

Background

IoT devices can send information from sensors using MQTT (or other communication protocols). The information is published to a particular topic and as the device is connected to the Internet, it can be collected by message brokers such as the one provided by AWS IoT. These messages can be stored in a database for later analysis or could theoretically be used to trigger a function that is created using AWS Lambda.

More complicated scenarios allow for messages to be collected by a local device (e.g. IoT Gateway device) allowing for situations where there may be intermittent Internet connectivity but always local connectivity. We are not going to explore this scenario in this lab.

Scenario: Your application will simulate a device that is going to send out the temperature every minute (you can configure this to be whatever time period you like) the data will be collected automatically into a DynamoDB table using a lambda function.

## [Step 1] Setup and configuration

Install the python AWS IoT SDK from here: <https://github.com/aws/aws-iot-device-sdk-python> or just use:

pip install AWSIoTPythonSDK

You are going to use this SDK for creating a client using MQTT

To configure the IoT device, follow the steps outlined in getting started with AWS IoT here:

<https://docs.aws.amazon.com/iot/latest/developerguide/iot-gs.html>

Don’t forget to do all of the steps with the certificate, policies etc.

Use your student number to identify the IoT thing. Configure it as TempSensor type of IoT and add it to the group CITS5503\_Temp\_Sensors

Create and download the certificates for your client

When attaching a policy, use the cits5530iotpolicy

## [Step 2] Test publish/subscribe

You can use the basicPubSub.py sample to modify to do the following below:

<https://github.com/aws/aws-iot-device-sdk-python/tree/master/samples/basicPubSub>

In the IoT console you can test publishing and subscribing to a topic – do that with your application to make sure that it is all working. As a topic, use

<student number>/temp

We are going to send a json payload of the format:

payload = {

"deviceId": "dgiot01",

"datetimestamp": "",

"temperature": ""

}

To send this, you will need to convert it to a string:

json.dumps(payload)

For datetimestamp you can create a timestamp using the function

datetime.datetime.now()

## [Step 3] Create a DynamoDB table to store events

Create a table in DynamoDB using the console with the table name <Student Number>IoTTable

The primary key will be deviceId as a string and the sort key will be datetimestamp as a string. Leave everything else as default.

## [Step 4] Create a lambda function to update table

Create a lambda function using the following code:

import boto3

import json

def lambda\_handler(event, context):

dynamodb = boto3.resource('dynamodb',

region\_name='ap-southeast-2',

endpoint\_url="https://dynamodb.ap-southeast-2.amazonaws.com")

table = dynamodb.Table('<YOUR TABLE NAME>')

response = table.put\_item(

Item={

'deviceId': event['deviceId'],

'temperature': event['temperature'],

'datetimestamp': event['datetimestamp']

}

)

return 'Success'

Select author from scratch and use the Python 3.6 runtime. Choose an existing role – lambda\_dynamodbFull\_role

You can test the function by creating a test event and making sure that the data goes into the DynamoDB table.

## [Step 5] Create an IoT Rule

In IoT Core, click on Act and create a rule. Give the rule a name with your student number and for the attribute put \* and Topic put the topic you are publishing temperature updates on

Add an action that is your lambda rule.

That should be it!

Run your publisher and see if the data goes into the DynamoDB table.

## Submission and Quiz

Submit the python file you wrote: respond to the quiz

## Respond to the Quiz

[1] An AWS lambda function allows code to be executed as long as you have an EC2 instance created to run it on

[A] True

[B] False

[2] AWS IoT uses what to authenticate IoT devices?

[A] An X.509 certificate and private key

[B] An X.509 certificate, public key and private key

[C] An X.509 certificate, public key and private key and AWS credentials for and IAM account access to the AWS API