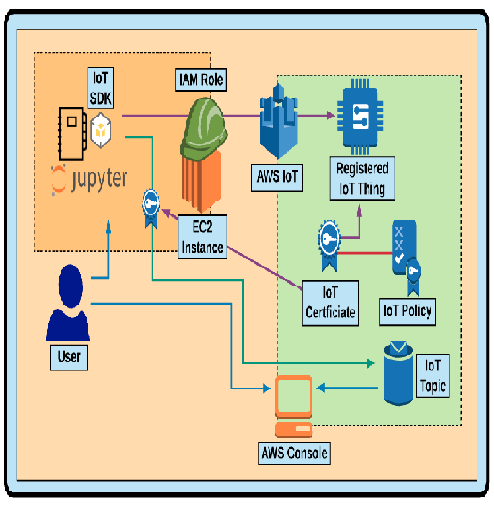
****

**Creating Things, Certificates, and Policies in the AWS Console**

1. Go to the IoT Section of the AWS console and create am IoT Thing
2. Create a new type to give this thing
3. Give the thing a group and potentially an attribute or two
4. Create a certificate for the thing and make sure to download all the certificates (and the first AWS certificate!)
5. Create a policy for the thing and attach it to the certificate

**Subscribe to a Topic**

1. In the Test section of the IoT console, subscribe to the topic penguinfeed/1
2. Publish a basic message to that same topic in the console

**Send data to the Topic using your "IoT Device"**

**Install the AWS IoT SDK for Python (this is not boto3!)**

**!**pip install AWSIoTPythonSDK

**Check that the SSL version is 1.0.1 or greater**

**import** ssl

ssl.OPENSSL\_VERSION

**Import some dependencies we'll use to send data to the Topic from the "IoT Device" of this EC2 Instance**

In [ ]:



**import** csv

**import** random

**import** json

**import** time

​

**from** AWSIoTPythonSDK.MQTTLib **import** AWSIoTMQTTClient

**from** time **import** sleep

**from** datetime **import** date, datetime

**Set some initial variables**

In [ ]:



CLIENT\_NAME **=** "penguin-sensor"

TOPIC **=** "penguinfeed/1"

​

*# Broker path is under AWS IoT > Settings (at the bottom left)*

*# Uncomment the next line after setting it*

*# BROKER\_PATH = "a2hyr29arhu1ac-ats.iot.us-east-1.amazonaws.com"*

​

*# RSA 2048 bit key: Amazon Root CA 1 found here:*

*# https://docs.aws.amazon.com/iot/latest/developerguide/managing-device-certs.html*

ROOT\_CA\_PATH **=** './AmazonRootCA1.pem'

​

*# These two keys are downloaded from the AWS IoT Console*

*# Upload them inside the Jupyter notebook and update/uncomment them*

*# PRIVATE\_KEY\_PATH = './14e764a6a2-private.pem.key'*

*# CERTIFICATE\_PATH = './14e764a6a2-certificate.pem.crt'*

**Create and Configure the IoT Client**

In [ ]:



IoTclient **=** AWSIoTMQTTClient(CLIENT\_NAME)

IoTclient.configureEndpoint(BROKER\_PATH, 8883)

IoTclient.configureCredentials(

ROOT\_CA\_PATH,

PRIVATE\_KEY\_PATH,

CERTIFICATE\_PATH

)

​

*# Allow the device to queue infinite messages*

IoTclient.configureOfflinePublishQueueing(**-**1)

​

*# Number of messages to send after a connection returns*

IoTclient.configureDrainingFrequency(2) *# 2 requests/second*

​

*# How long to wait for a [dis]connection to complete (in seconds)*

IoTclient.configureConnectDisconnectTimeout(10)

​

*# How long to wait for publish/[un]subscribe (in seconds)*

IoTclient.configureMQTTOperationTimeout(5)

​

​

IoTclient.connect()

IoTclient.publish(TOPIC, "connected", 0)

**Create and Send Payloads to the IoT Topic**

In [ ]:



**def** create\_payload():

penguins **=** [

"Adelie", "African", "Chinstrap", "Emperor",

"Erect-crested", "Fiordland", "Galapagos",

"Gentoo", "Humboldt", "King", "Little",

"Macaroni", "Magellanic", "Rockhopper",

"Royal", "Snares", "Yellow-eyed"

]

payload **=** json.dumps({

"penguin\_type": random.choice(penguins),

"latitude": random.uniform(**-**90, 90),

"longitude": random.uniform(**-**180,180)

})

**return** payload

​

**while** **True**:

IoTclient.publish(TOPIC, create\_payload(), 0)

**All Done! Awesome Job!**