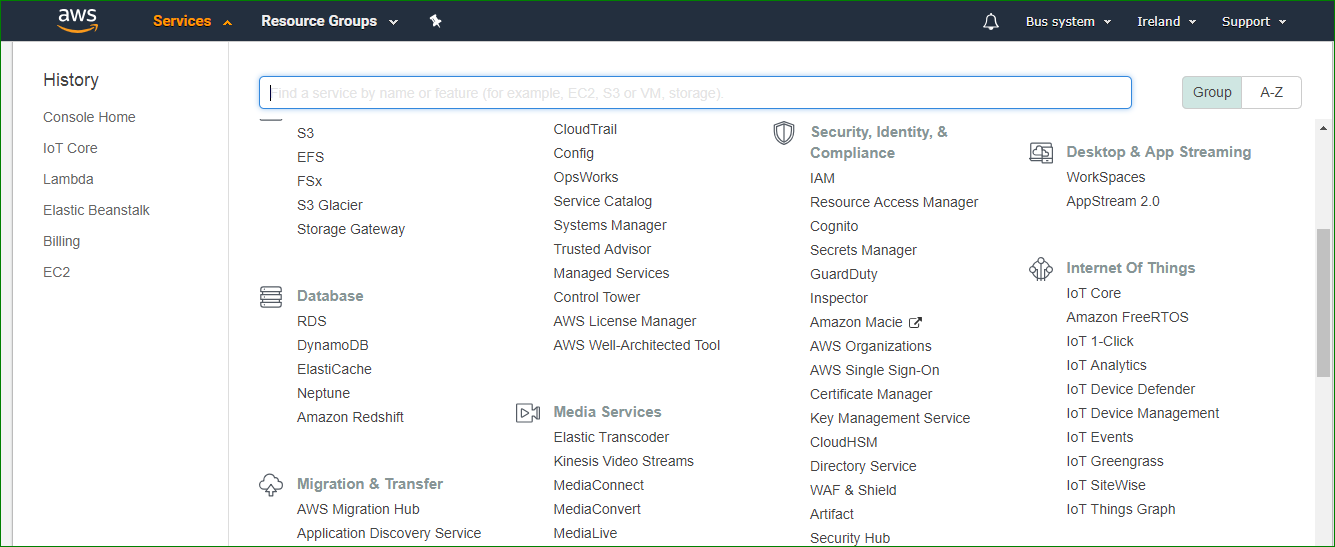
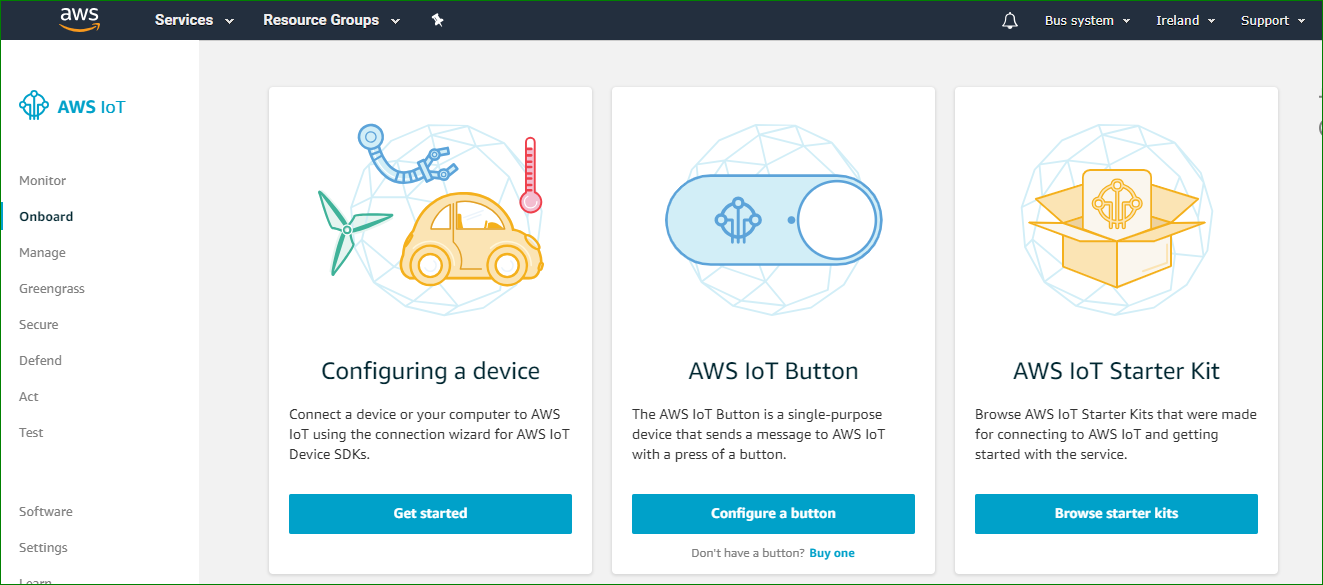
**Connect raspberry pi to aws iot using mqtt**

**Steps**

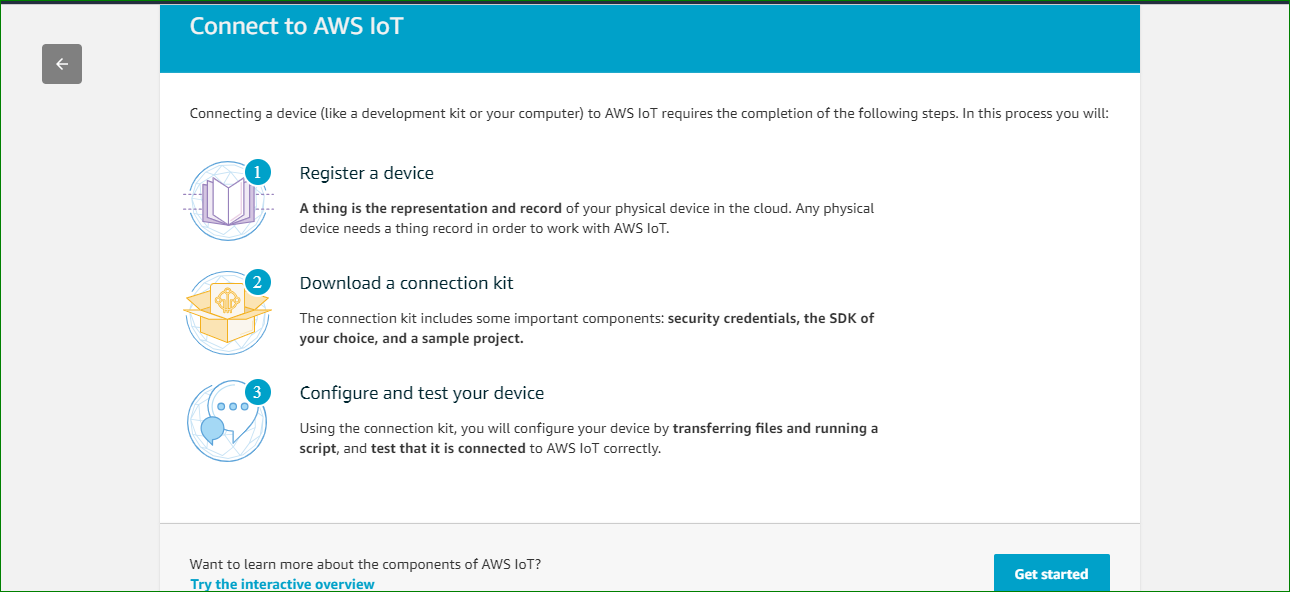
1. Login to Amazon AWS cloud
2. Services -> select iot core

****

3. → onboard



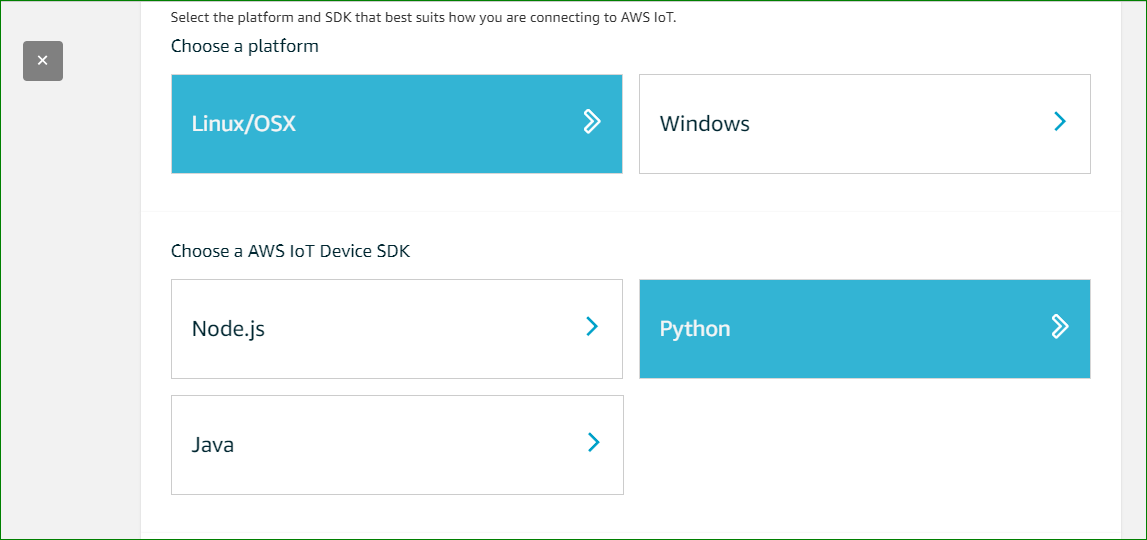
4. Get started



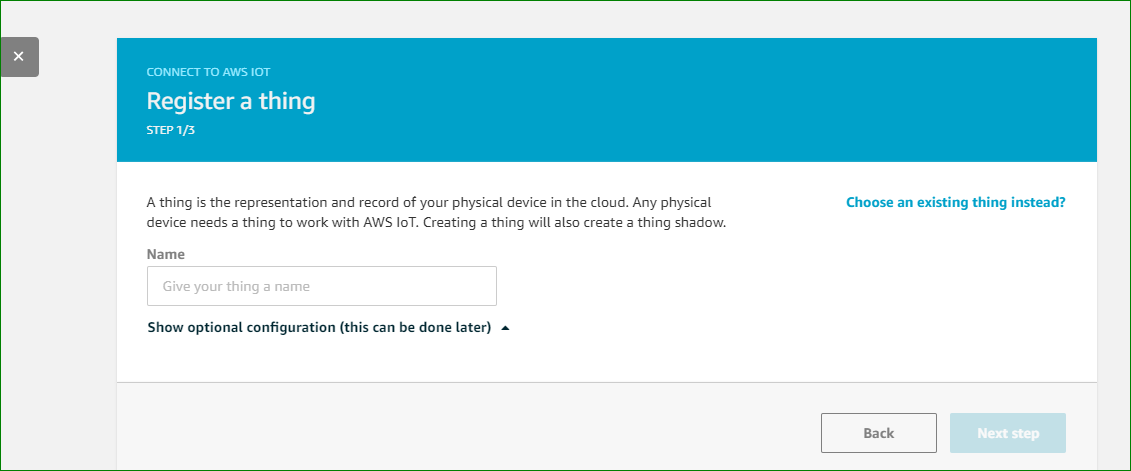
5. Select

Choose a platform - Linux

Choose a AWS IoT Device SDK - Python



→ Next

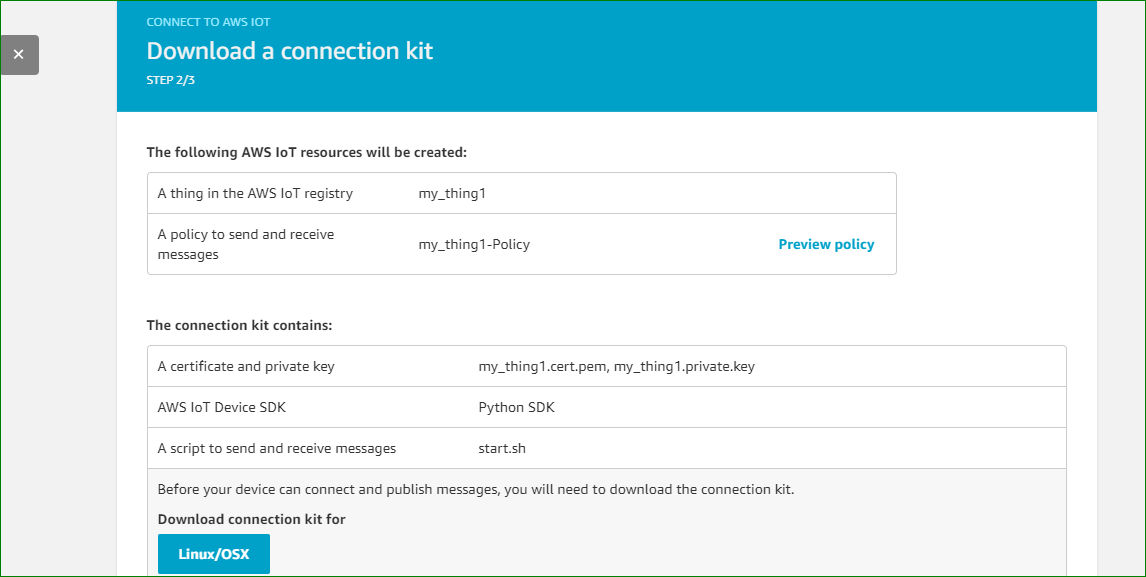


Name → my\_thing1

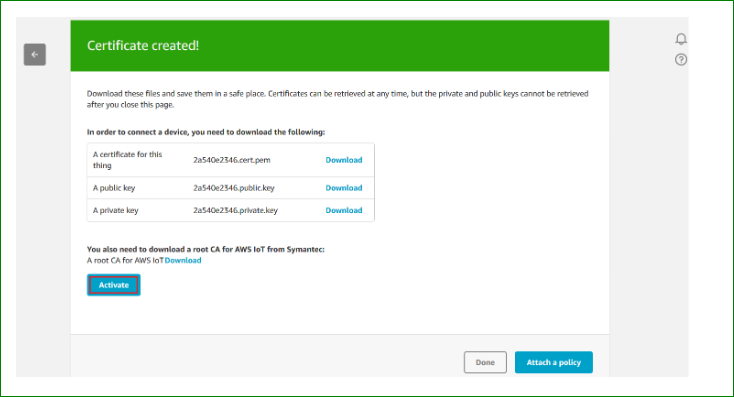
You can give any name to your thing

→ next Step

6. Download the sdk



7. Download the certificates



8. Click on activate

9. Download the AWSiotpythonsdk using the command

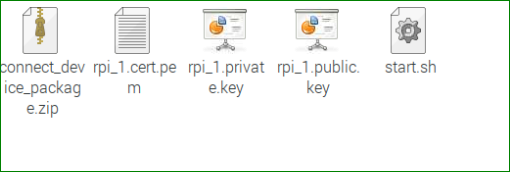
**$ pip install AWSIoTPythonSDK**

10. There are 2 steps to download the root certificate

i)

Unzip the downloaded file

U will have the following files after unzipping



Give execute permissions to the start.sh file

By typing command in the command line

**$ sudo +chmod ./start.sh**

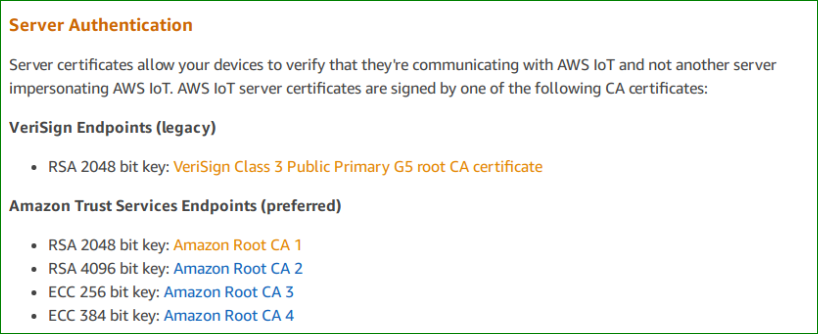
Execute the start.sh file

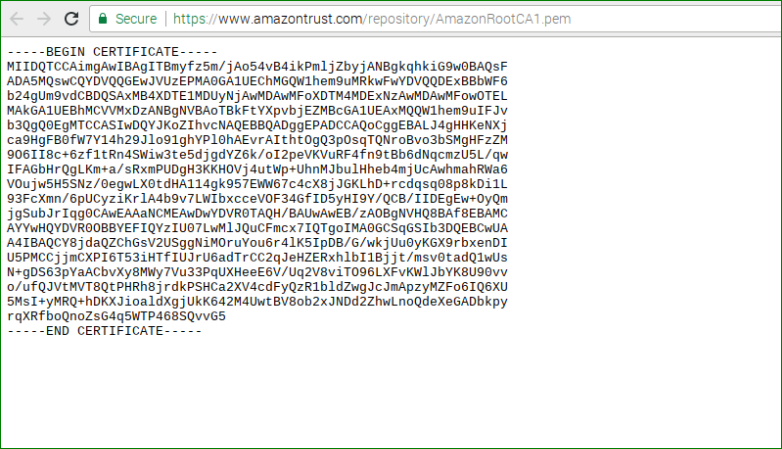
**$ ./start.sh**

ii) if the root certificate is not downloaded then go to the link

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

Select Amazon Root CA 1





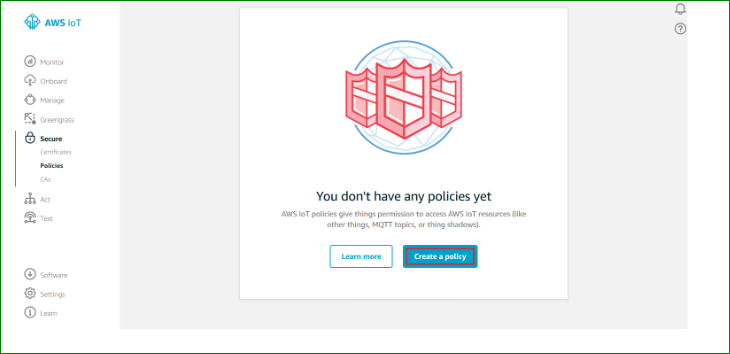
Copy the contents and create a new file in the folder where all the other keys are dowloaded

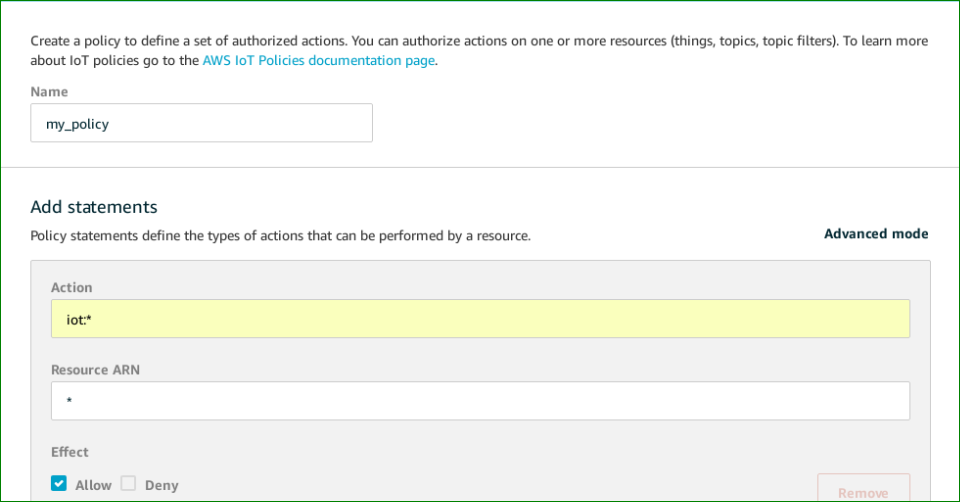
Save the file as root-CA.crt extension

**Note: all the keys and the root certificate should be one folder.**

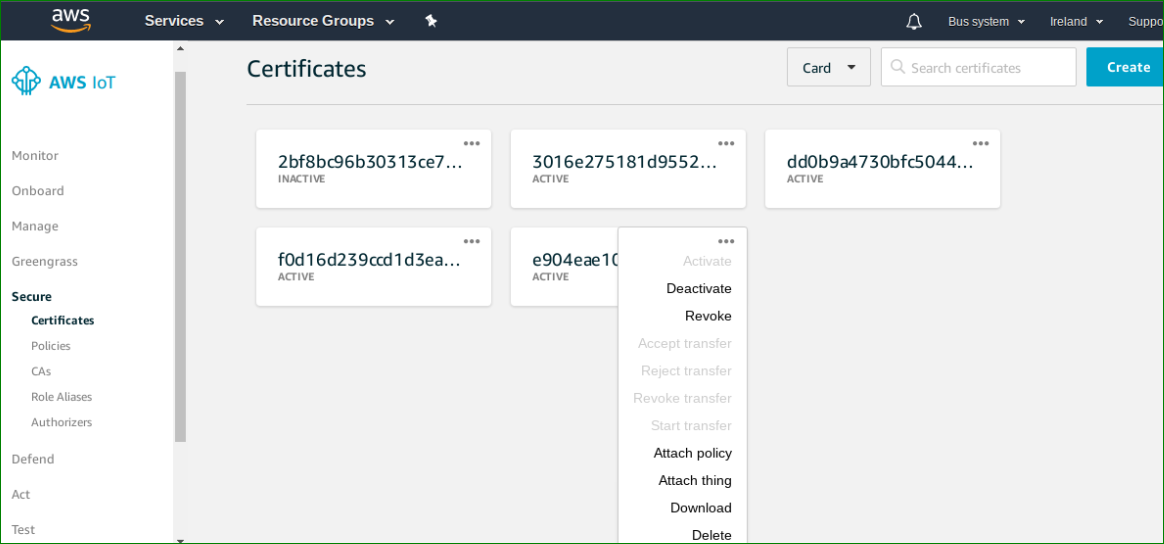
**To create an AWS IoT policy:**

**In the left navigation pane, choose Secure, and then Policies. On the You don't have a policy yet page, choose Create a policy.**

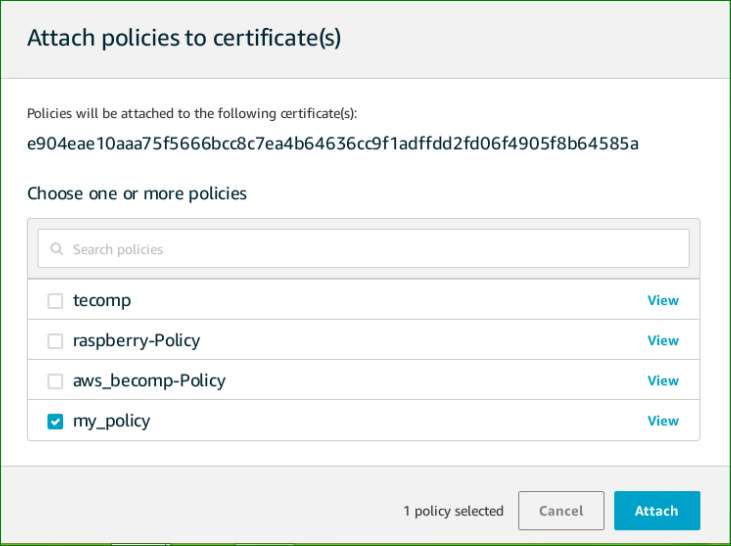
****



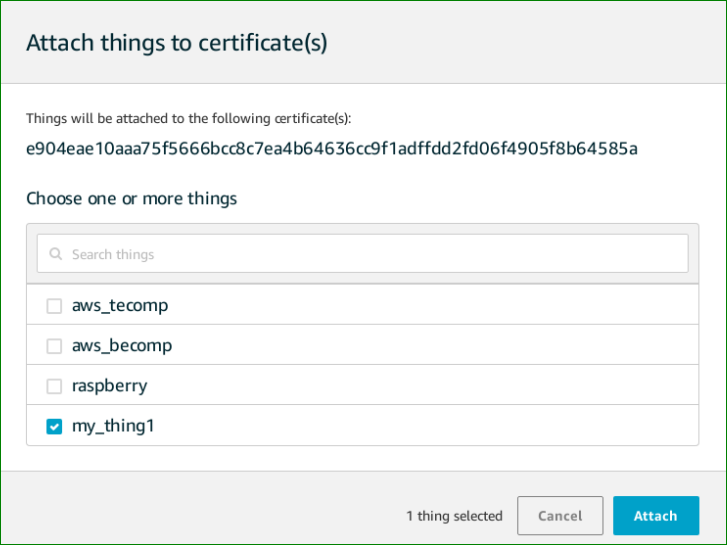
secure→ certificate → right click on certificate → attach a policy



Attach the policy to certificate



Attach the certificate to the registered thing



Save the following two python scripts in the same folder where keys are downloaded

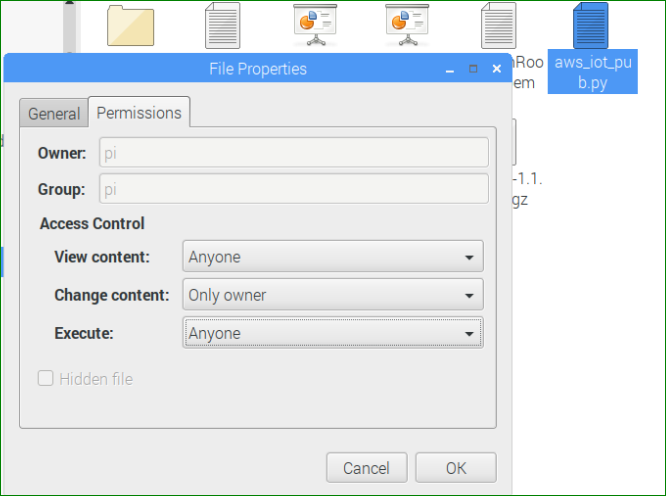
Aws\_iot\_pub.py

#!/usr/bin/python  
  
############################################   
# PROBLEM STATEMENT:  
# This program will publish test mqtt messages using the AWS IOT hub  
#   
# To test this program you have to run first its companinon aws\_iot\_sub.py  
# that will subscribe and show all the messages sent by this program  
#  
############################################  
  
############################################  
# STEPS:  
#  
# 1. Sign in to AWS Amazon > Services > AWS IoT > Settings > copy Endpoint  
# This is your awshost  
#   
# 2. Change following things in the below program:  
# a. awshost (from step 1)  
# b. clientId (Thing\_Name)  
# c. thingName (Thing\_Name)  
# d. caPath (root-CA\_certificate\_Name)  
# e. certPath (<Thing\_Name>.cert.pem)  
# f. keyPath (<Thing\_Name>.private.key)  
#   
# 3. Paste aws\_iot\_pub.py & aws\_iot\_sub.py python scripts in folder where all unzipped aws files are kept.   
# 4. Provide Executable permition for both the python scripts.  
# 5. Run aws\_iot\_sub.py script  
# 6. Run this aws\_iot\_pub.py python script  
#  
############################################  
  
# importing libraries  
import paho.mqtt.client as paho  
import os  
import socket  
import ssl  
from time import sleep  
from random import uniform  
   
connflag = False  
   
def on\_connect(client, userdata, flags, rc): # func for making connection  
 global connflag  
 print "Connected to AWS"  
 connflag = True  
 print("Connection returned result: " + str(rc) )  
   
def on\_message(client, userdata, msg): # Func for Sending msg  
 print(msg.topic+" "+str(msg.payload))  
   
#def on\_log(client, userdata, level, buf):  
# print(msg.topic+" "+str(msg.payload))  
   
mqttc = paho.Client() # mqttc object  
mqttc.on\_connect = on\_connect # assign on\_connect func  
mqttc.on\_message = on\_message # assign on\_message func  
#mqttc.on\_log = on\_log  
  
#### Change following parameters ####   
awshost = "xxxxxxxxxxxxxxxxxxx.amazonaws.com" # Endpoint  
awsport = 8883 # Port no.   
clientId = "thing\_name" # Thing\_Name  
thingName = "thing\_name" # Thing\_Name  
caPath = "/home/pi/Downloads/AmazonRootCA1.pem" # Root\_CA\_Certificate\_Name  
certPath = "/home/pi/Downloads/3016e27518-certificate.pem.crt" # <Thing\_Name>.cert.pem  
keyPath = "/home/pi/Downloads/3016e27518-private.pem.key" # <Thing\_Name>.private.key  
   
mqttc.tls\_set(caPath, certfile=certPath, keyfile=keyPath, cert\_reqs=ssl.CERT\_REQUIRED, tls\_version=ssl.PROTOCOL\_TLSv1\_2, ciphers=None) # pass parameters  
   
mqttc.connect(awshost, awsport, keepalive=60) # connect to aws server  
   
mqttc.loop\_start() # Start the loop  
   
while 1==1:  
 sleep(5)  
 if connflag == True:  
 tempreading = uniform(20.0,25.0) # Generating Temperature Readings   
 mqttc.publish("temperature", tempreading, qos=1) # topic: temperature # Publishing Temperature values  
 print("msg sent: temperature " + "%.2f" % tempreading ) # Print sent temperature msg on console  
 else:  
 print("waiting for connection...")

Aws\_iot\_sub.py

#!/usr/bin/python  
  
############################################   
# PROBLEM STATEMENT:  
#  
# This program will subscribe and show all the messages sent by its companion   
# aws\_iot\_pub.py using AWS IoT hub  
#  
############################################  
  
############################################  
# STEPS:  
#  
# 1. Sign in to AWS Amazon > Services > AWS IoT > Settings > copy Endpoint  
# This is your awshost  
#   
# 2. Change following things in the below program:  
# a. awshost (from step 1)  
# b. clientId (Thing\_Name)  
# c. thingName (Thing\_Name)  
# d. caPath (root-CA\_certificate\_Name)  
# e. certPath (<Thing\_Name>.cert.pem)  
# f. keyPath (<Thing\_Name>.private.key)  
#   
# 3. Paste aws\_iot\_pub.py & aws\_iot\_sub.py python scripts in folder where all unzipped aws files are kept.   
# 4. Provide Executable permition for both the python scripts.  
# 5. Run aws\_iot\_sub.py script  
# 6. Run this aws\_iot\_pub.py python script  
#  
############################################  
  
# importing libraries  
import paho.mqtt.client as paho  
import os  
import socket  
import ssl  
   
def on\_connect(client, userdata, flags, rc): # func for making connection  
 print("Connection returned result: " + str(rc) )  
 # Subscribing in on\_connect() means that if we lose the connection and  
 # reconnect then subscriptions will be renewed.  
 client.subscribe("#" , 1 ) # Subscribe to all topics  
   
def on\_message(client, userdata, msg): # Func for receiving msgs  
 print("topic: "+msg.topic)  
 print("payload: "+str(msg.payload))  
   
#def on\_log(client, userdata, level, msg):  
# print(msg.topic+" "+str(msg.payload))  
   
mqttc = paho.Client() # mqttc object  
mqttc.on\_connect = on\_connect # assign on\_connect func  
mqttc.on\_message = on\_message # assign on\_message func  
#mqttc.on\_log = on\_log  
  
#### Change following parameters ####   
awshost = "xxxxxxxxxxxxxxxxxxxxxxxxxxxx.amazonaws.com" # Endpoint  
awsport = 8883 # Port no.   
clientId = "thing\_name" # Thing\_Name  
thingName = "thing\_name" # Thing\_Name  
caPath = "/home/pi/Downloads/AmazonRootCA1.pem" # Root\_CA\_Certificate\_Name  
certPath = "/home/pi/Downloads/3016e27518-certificate.pem.crt" # <Thing\_Name>.cert.pem  
keyPath = "/home/pi/Downloads/3016e27518-private.pem.key" # <Thing\_Name>.private.key  
 # <Thing\_Name>.private.key  
   
mqttc.tls\_set(caPath, certfile=certPath, keyfile=keyPath, cert\_reqs=ssl.CERT\_REQUIRED, tls\_version=ssl.PROTOCOL\_TLSv1\_2, ciphers=None) # pass parameters  
   
mqttc.connect(awshost, awsport, keepalive=60) # connect to aws server  
   
mqttc.loop\_forever() # Start receiving in loop

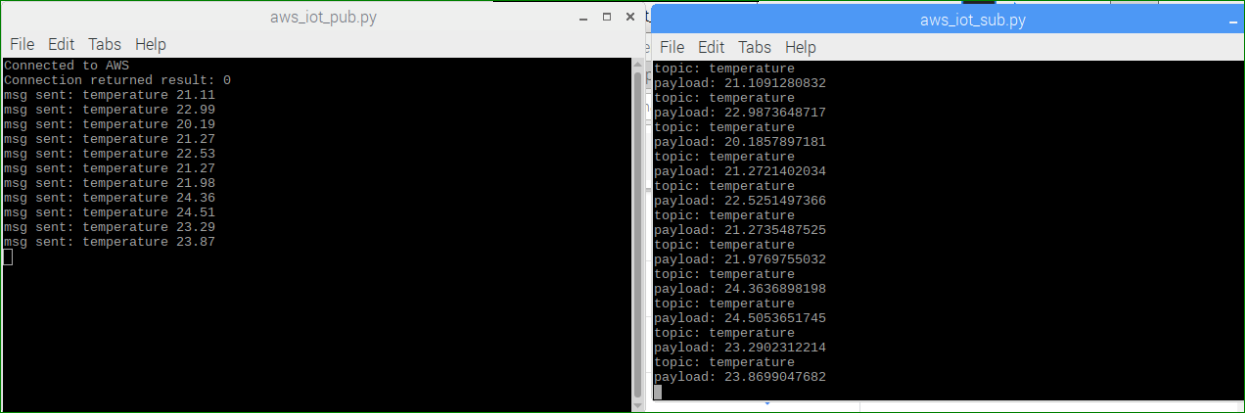
Give execute permission to both the files



Change excute to → Anyone

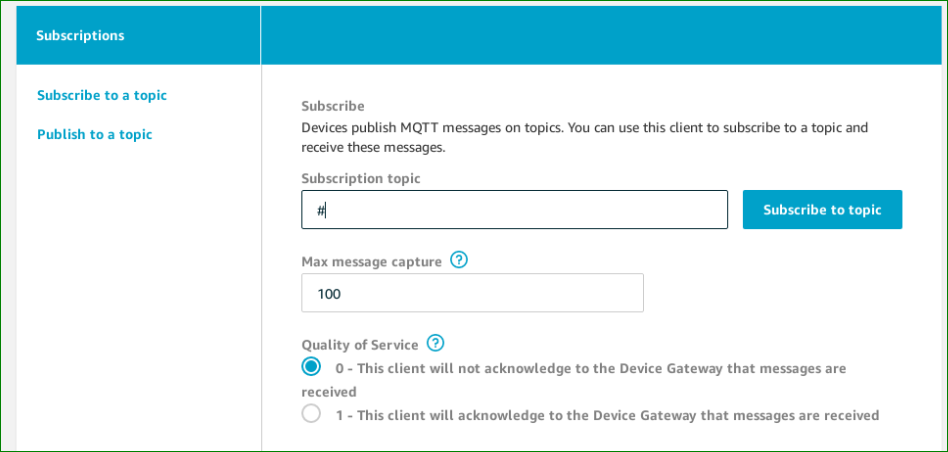
Do it for both the files

Run both files



To check on the aws iot

In the navigation pane → Test →



Subscribe to all the topics i.e. #

Click subscribe

