```
 #!/usr/bin/python3

#required libraries
import sys
import ssl
import json
import paho.mqtt.client as mqtt

#called while client tries to establish connection with the server
def on_connect(mqttc, obj, flags, rc):
    if rc==0:
        print ("Subscriber Connection status code: "+str(rc)+" | Connection status: successful")
        mqttc.subscribe("$aws/things/pi_assn4/shadow/update/accepted", qos=0)

        mqttc.publish("$aws/things/pi_assn4/shadow/update",'{"state":{"reported":{"color":"Fu"}}}
        ')
    elif rc==1:
        print ("Subscriber Connection status code: "+str(rc)+" | Connection status: Connection
        refused")

#called when a topic is successfully subscribed to
def on_subscribe(mqttc, obj, mid, granted_qos):
    print("Subscribed: "+str(mid)+" "+str(granted_qos)+"data"+str(obj))

#called when a message is received by a topic
def on_message(mqttc, obj, msg):
    print("Received message from topic: "+msg.topic+" | QoS: "+str(msg.qos)+" | Data Received:
    "+str(msg.payload))

#creating a client with client-id=mqtt-test
mqttc = mqtt.Client(client_id="cgao")

mqttc.on_connect = on_connect
mqttc.on_subscribe = on_subscribe
mqttc.on_message = on_message

#Configure network encryption and authentication options. Enables SSL/TLS support.
#adding client-side certificates and enabling tlsv1.2 support as required by aws-iot service
mqttc.tls_set(ca_certs="/home/pi/Desktop/rootCA.pem.crt",
          certfile="/home/pi/Downloads/7409fdbcb2-certificate.pem.crt",
          keyfile="/home/pi/Downloads/7409fdbcb2-private.pem.key",
              tls_version=ssl.PROTOCOL_TLSv1_2,
              ciphers=None)

#connecting to aws-account-specific-iot-endpoint
mqttc.connect("a1qup6k0p06lhy.iot.us-west-2.amazonaws.com", port=8883) #AWS IoT service
hostname and portno

#automatically handles reconnecting
mqttc.loop_forever()
```