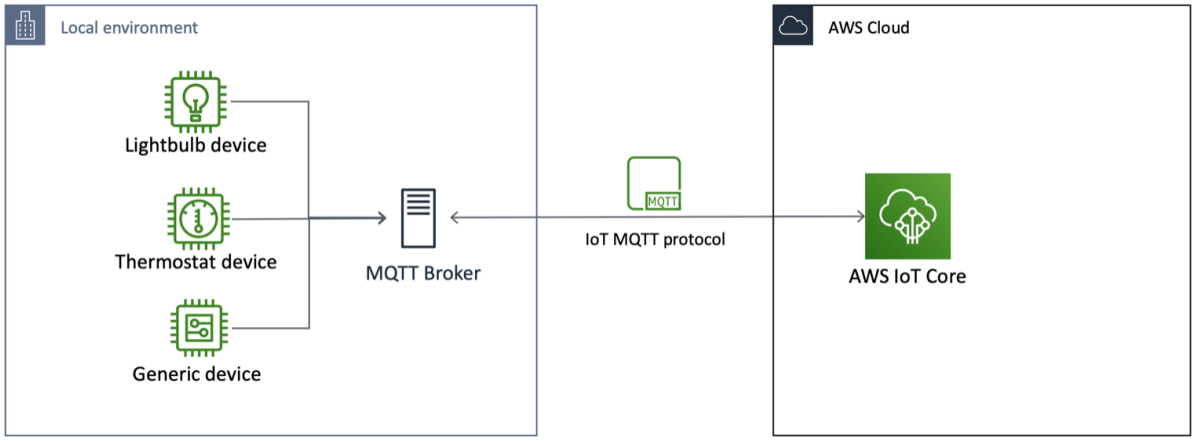
# Tutorial 6 - Communicate to AWS IoT Core

## Aim

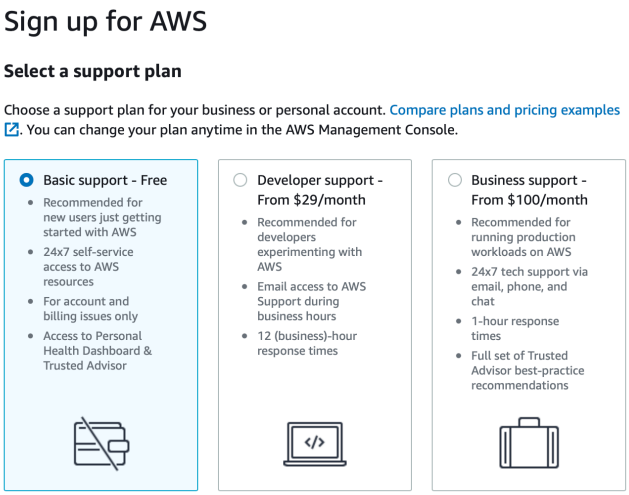
This tutorial will show you how to implement the 'Bridge' capability to setup bi-directional exchange of data with AWS IoT Core through MQTT messages. This will enable your devices to communicate locally with the Mosquitto broker/other SDKs and with AWS IoT Core to benefit from the power of the AWS Cloud.




## Configure the Bridge to AWS IoT

### Create Your AWS Account

If you do not have an AWS account, please go to https://aws.amazon.com to create a free personal root user account. This course will not use any charge services, so the basic support plan is sufficient.

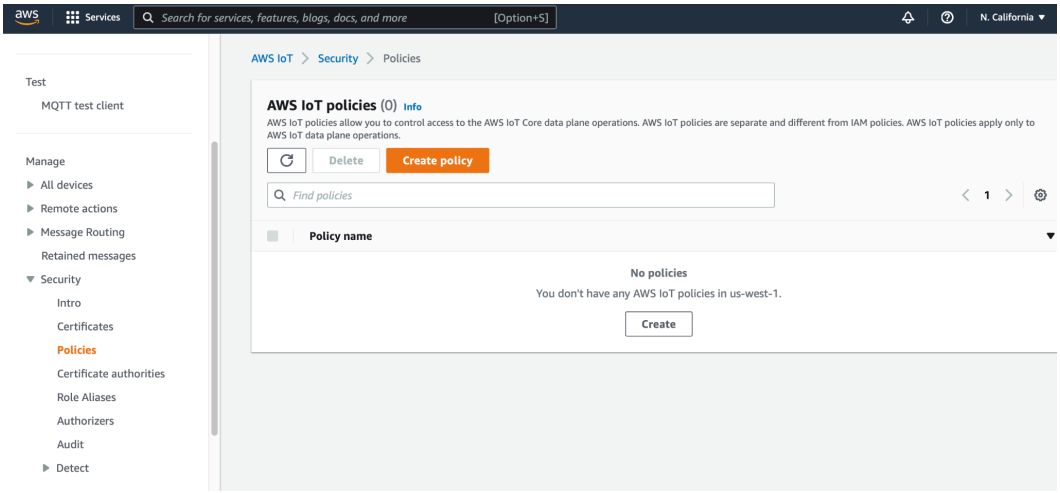



Once you have received the activation email, your aws account is ready for use.

## Create the resources on AWS IoT

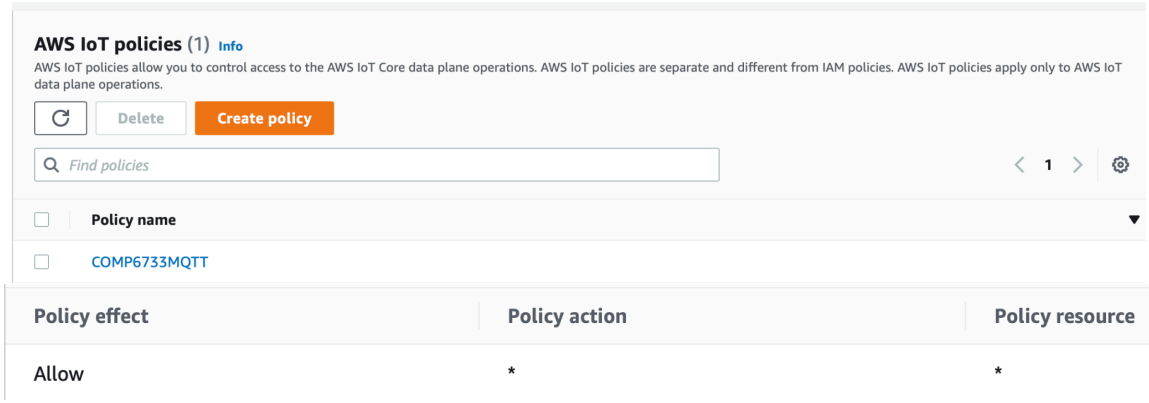
Now we will need to configure the bridge so that we can create a bi-directional connection to AWS IoT Core. We will first use the AWS CLI to create the necessary resources on AWS IoT side.

1. Search **IoT core** on the top navigation bar, enter your AWS IoT page. Then, expand the **Security** option on the left side and click **Policies**, now you will be in the page showing below:

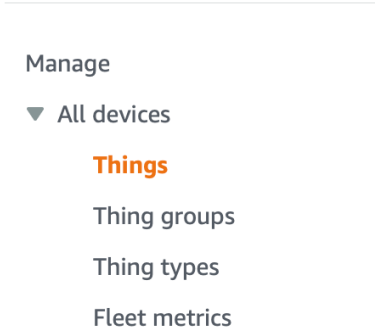



2. Now click **Create Policy**, enter the policy name (e.g. COMP6733MQTT). Under policy statement, set **policy effect** to **allow**, **policy action** and **policy resource** to ***.
   (Note: Allowing all AWS IoT action (iot:*) is useful for testing but it's a best practice to increase security for a production setup)

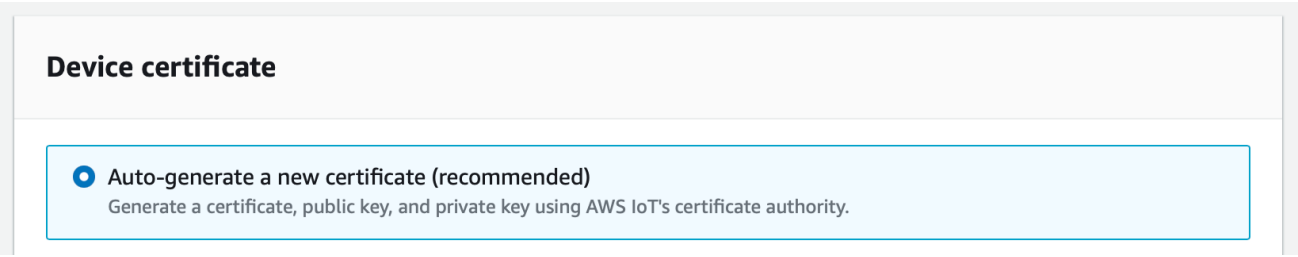
   Save the policy and now you can see your policy show on the page.

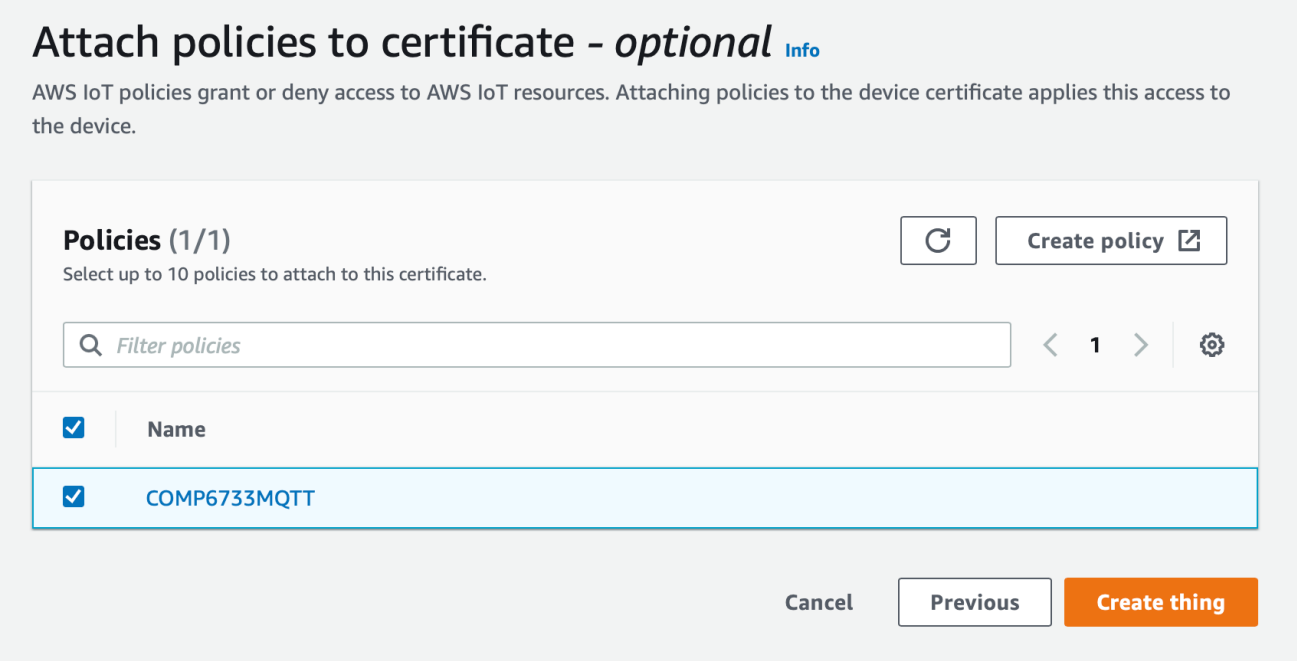



3. Next step, we are going to create a Thing. Expand **All devices**, then go the **Things** page and create a new thing.

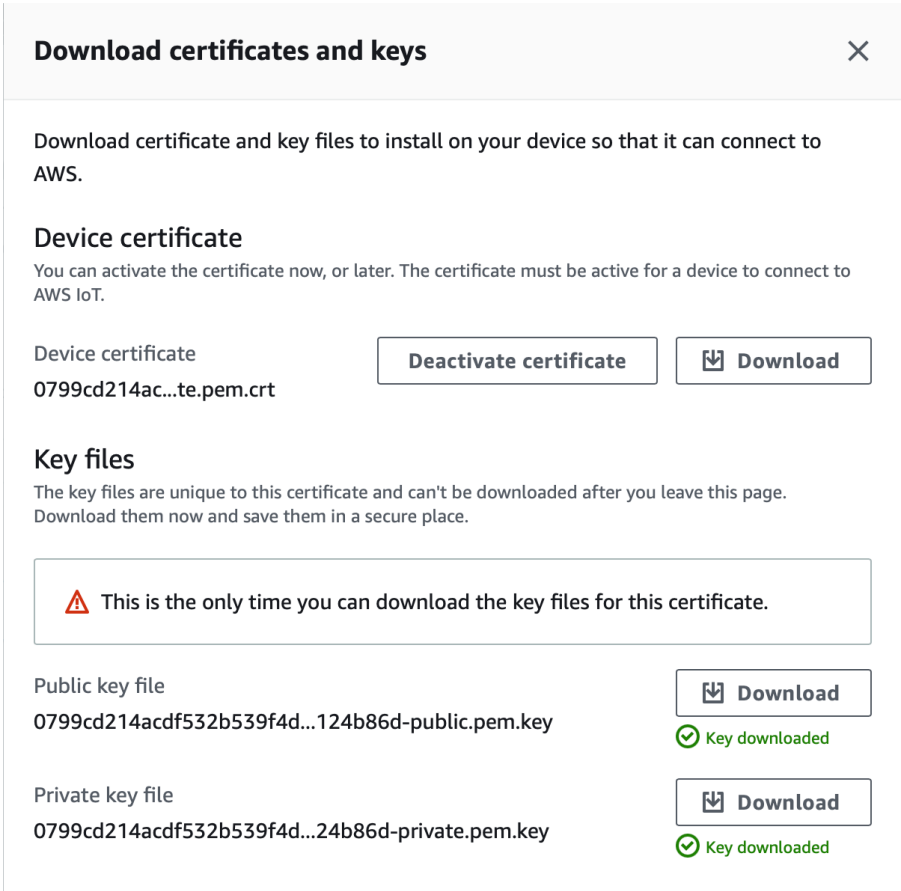



4. Enter your thing name (e.g. COMP6733Thing), then keep other settings as default and go the next page: **Device certificate**. Please select the recommended one. In the next step, attach the policy that you just created (e.g. COMP6733MQTT) and click **Create thing**.

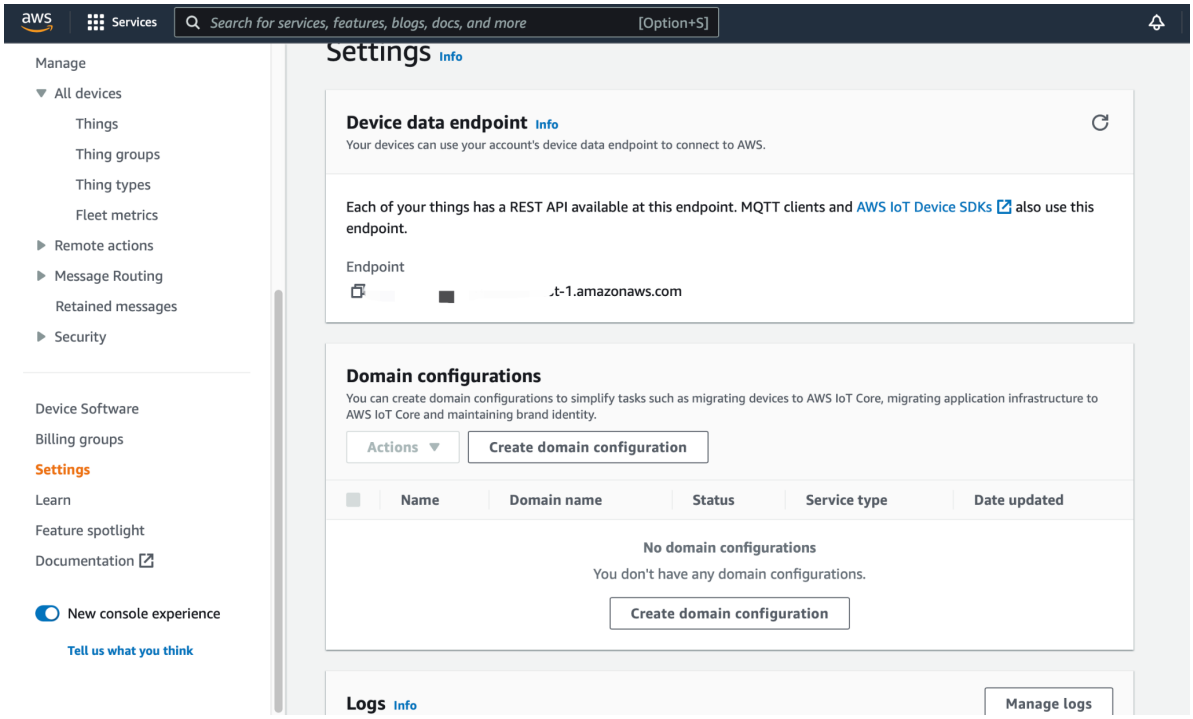





5. After you creating your thing, AWS will allow you to download certificates and keys. **This is very important! This is the only time you can download these files.** Please download your device certificate xxx.crt and your private and public key files, as well as the RootCA.pem file. Now you should have 4 files in your local directory.




6. In the last step, navigate to the **Settings** page on AWS IoT. You can find your unique **Endpoint** (e.g. xxxxxx.amazonnaws.com).




# Sending Message to AWS IoT Core

With the files you obtained from previous steps, you are now ready to send the message to AWS IoT from your device.

You can configure the Mosquitto Bridge to AWS IoT with these files by looking at the tutorial: Bridge Mosquitto MQTT Broker to AWS IoT to publish/subscribe messages. However, we will introduce another method here, using an python sdk:

Using pip3 to install

```
pip3 install AWSIoTPythonSDK
```

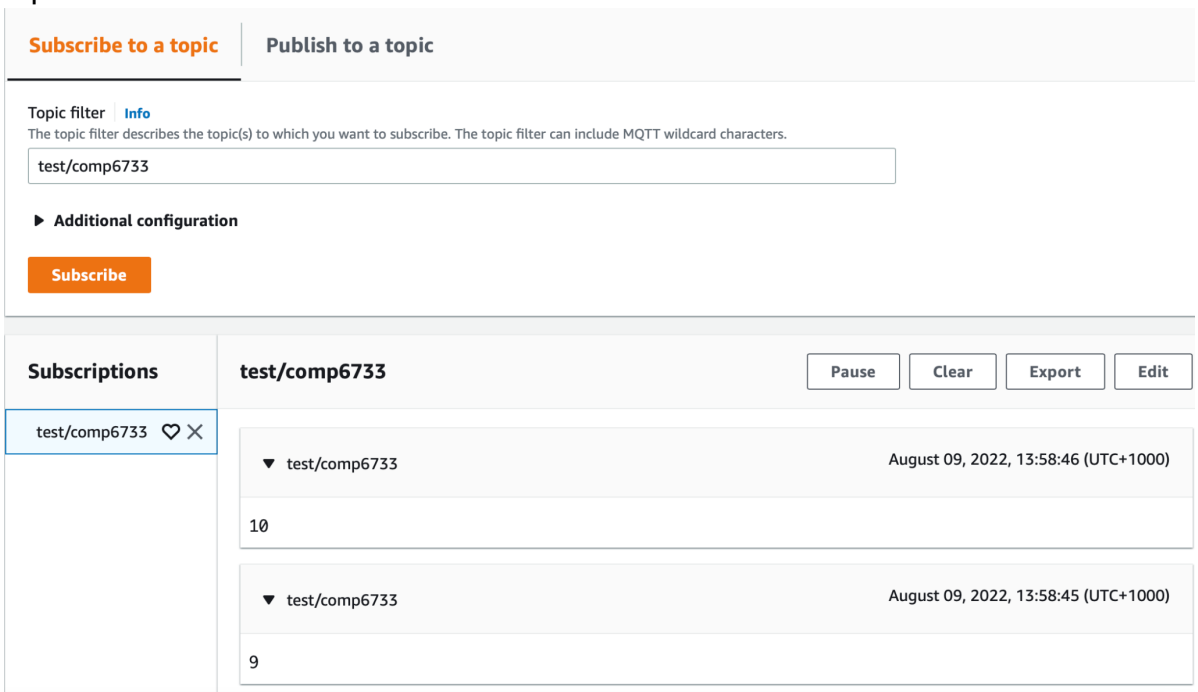
Then, create a .py code under the same folder of your credential files, copy the following python3 code and change endpoint address, credentials file name to yours.

```
import time as t
import json
import AWSIoTPythonSDK.MQTTLib as AWSIoTpyMQTT

# Client configuration with endpoint and credentials
myClient = AWSIoTpyMQTT.AWSIoTMQTTClient("testDevice")
myClient.configureEndpoint('xxxx.amazonaws.com',8883)
myClient.configureCredentials("AmazonRootCA1.pem","0799cd214acdf532b539f4d0ecee9d00d1ac
2be367d04939753e3bead124b86d-
private.pem.key","0799cd214acdf532b539f4d0ecee9d00d1ac2be367d04939753e3bead124b86d-
```

```
certificate.pem.crt")

myClient.connect()

for i in range(10):
   message = str(i+1)
   myClient.publish("test/comp6733",message,1)
   print("Published: '"+ message + " to test/comp6733")
   t.sleep(0.5)

myClient.disconnect()
```

Go to your AWS IoT **MQTT test client** page, and subscribe the topic we used in the code: test/comp6733.


Now run your python code, you should see 10 messages sending to your subscription topic as below:



# Install Mosquitto MQTT Broker

Typically, you should install this on what you view as your local gateway which is the device that will be the link between your local devices and other local devices or to the AWS Cloud.

One way to install mosquitto into your own Linux machine is by typing the following command lines:

```
#update the ca-cert

sudo apt-get install --reinstall ca-certificates
sudo update-ca-certificates

#Update the list of repositories with one containing the latest version of #Mosquitto and update the package lists

sudo apt-add-repository ppa:mosquitto-dev/mosquitto-ppa

sudo apt-get update

#Install the Mosquitto broker, Mosquitto clients

sudo apt-get install mosquitto

sudo apt-get install mosquitto-clients
```

Now you should be able to test the mosquitto broker locally.

In separate terminal windows do the following:

Start the broker on port 5000 (by default is 1883 but it might already in use):

```
mosquitto -p 5000
```

Start the command line subscriber:

```
mosquitto_sub -v -t 'test/topic' -p 5000
```

Publish test message with the command line publisher:

```
mosquitto_pub -t 'test/topic' -m 'helloCOMP6733' -p 5000
```

As well as seeing both the subscriber and publisher connection messages in the broker terminal the following should be printed in the subscriber terminal:

```
test/topic helloCOMP6733
```

You can also try on different OS such as Windows and MacOS.

## Conclusion

This lesson introduce how to setup and configure your AWS account, and communicate via MQTT from your own device. For more information please refer to the related documentation.

## Related Document

- Bridge Mosquitto MQTT Broker to AWS IoT