AW-CU427-P

IoT Connectivity Module for AWS IoT Core

Getting Started Guide

Rev. 0.2

Revision History

Version	Revision Date	Description	Initials	Approved
0.1	2021/02/04	Initial version	Steven Jian Jackson Boon	N.C. Chen S.C Chueh
0.2	2021/10/28	Add introduction of “Over-the-Air Updates”	Josh Lin	Patrick Lin

Table of Contents

1. Introduction of AW-CU427-P 4

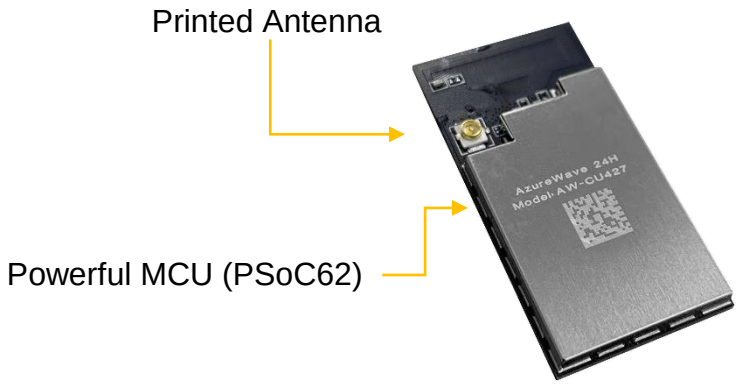
1.1 Product Overview 4
1.2 Block Diagram 5
1.3 Schematics 6
1.4 Pin Definition 7
1.5 Layout Guide and SMT Process Notification 10

2. AWS Command Example 11

2.1 Getting Started with AWS IoT Core 11
2.2 Publish and Monitor MQTT message on the cloud 15
2.3 Over-the-Air Updates 17

1. Introduction of AW-CU427-P

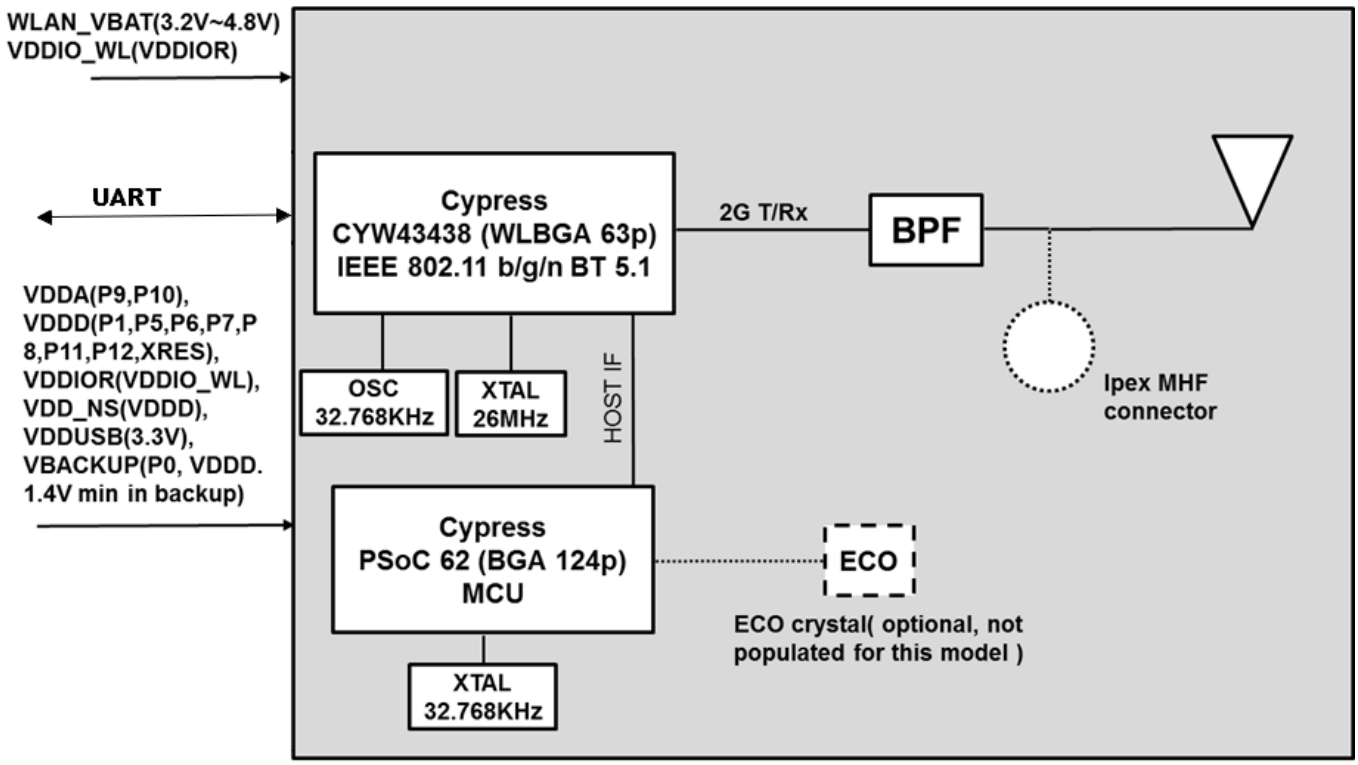
1.1 Product Overview



AW-CU427-P is a Wi-Fi Module with FreeRTOS qualified MCU that uses AT commands to securely and efficiently communicate with AWS IoT Core

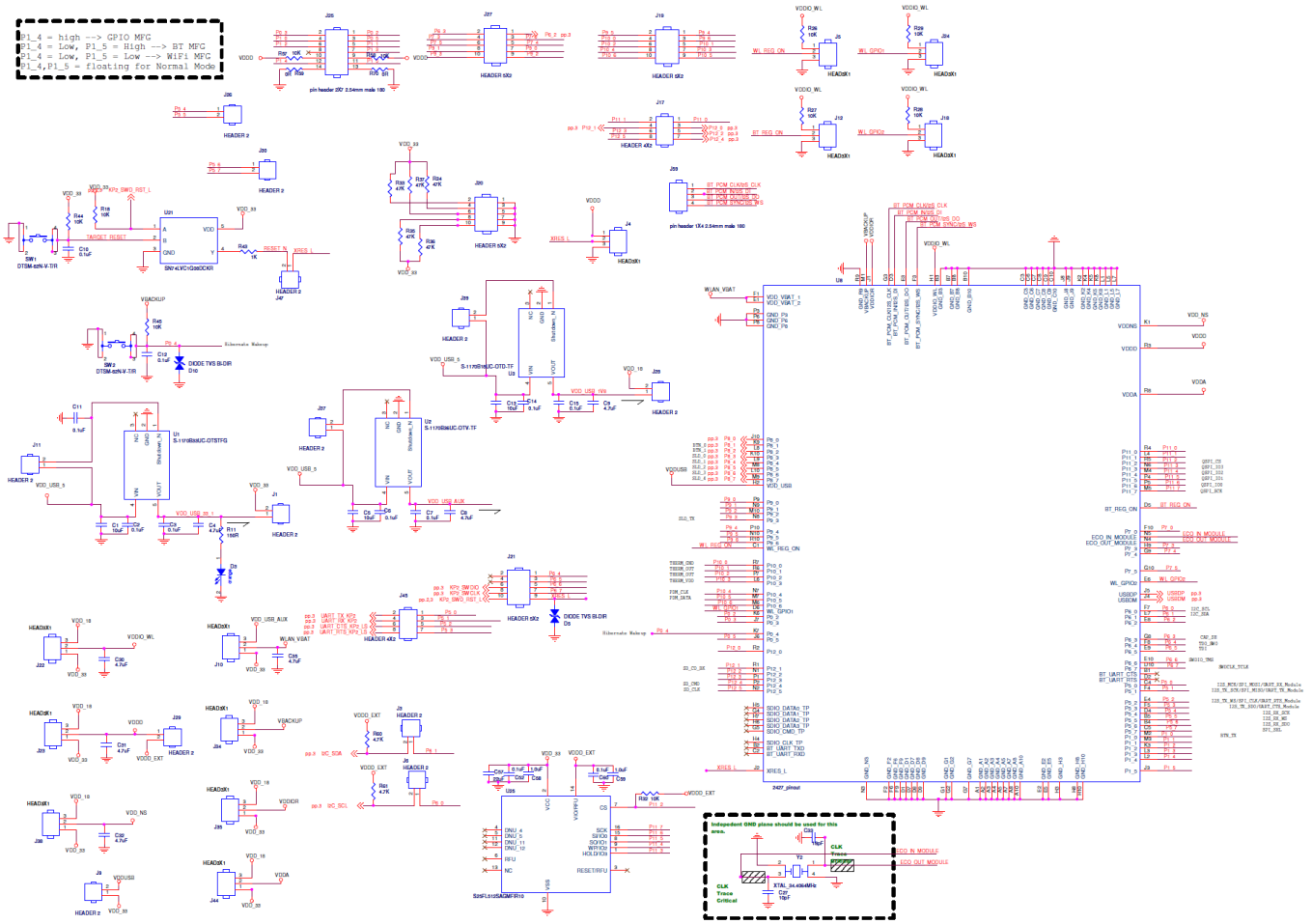
- Hardware specification defined by Amazon and AzureWave
- With AICM, end-device become AWS IoT Device
- UART interface for end-device to connect with
- Rich AT commands for end-device to communicate with AWS IoT Core

1.2 Block Diagram



1.3 Schematics

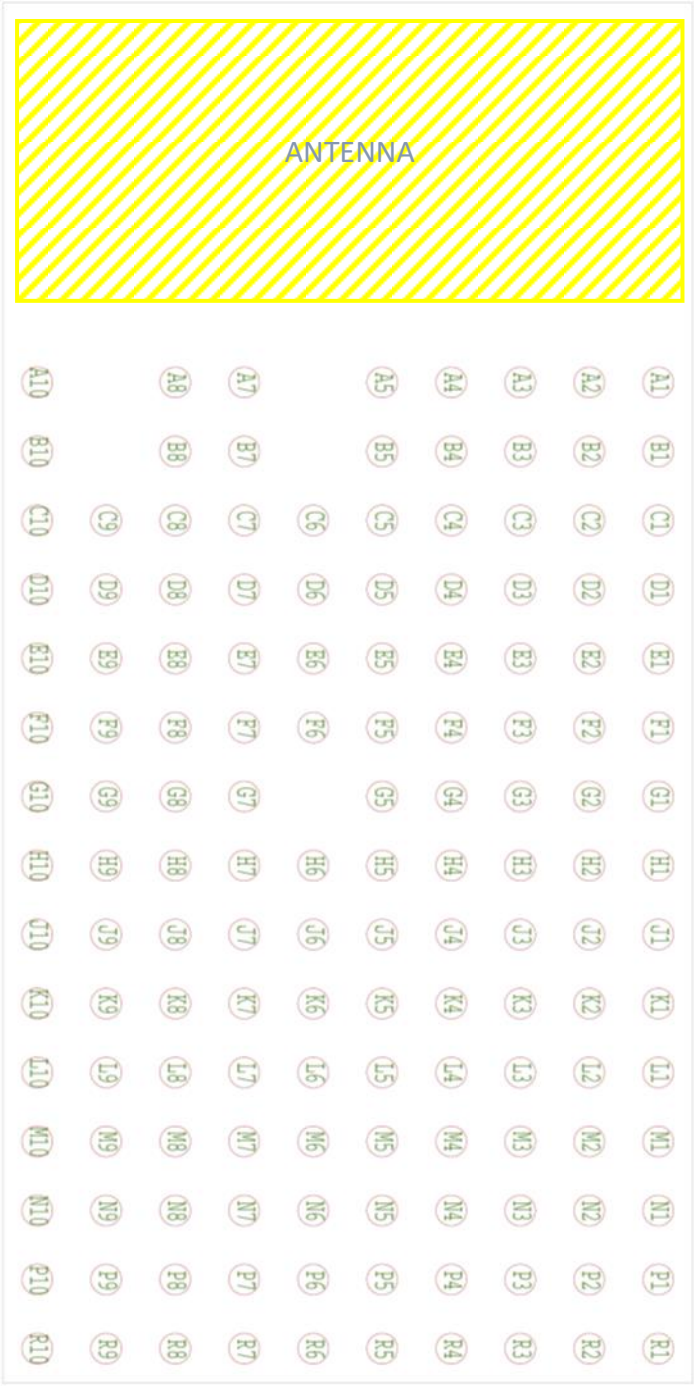
Module pinout for AW-CU427-P



1.4 Pin Definition

Pin Map

AW-CU427-P Top View Pin Map



Pin Table

Pin No	Definition	Basic Description	Voltage	Type
A1	GND_A1	Ground.		GND
A10	GND_A10	Ground.		GND
A2	GND_A2	Ground.		GND
A3	GND_A3	Ground.		GND
A4	GND_A4	Ground.		GND
A5	GND_A5	Ground.		GND
A7	GND_A7	Ground.		GND
A8	GND_A8	Ground.		GND
B10	GND_B10	Ground.		GND
B3	GND_B3	Ground.		GND
B7	GND_B7	Ground.		GND
B8	GND_B8	Ground.		GND
C10	GND_C10	Ground.		GND
C3	GND_C3	Ground.		GND
C6	GND_C6	Ground.		GND
C7	GND_C7	Ground.		GND
C8	GND_C8	Ground.		GND
C9	GND_C9	Ground.		GND
D1	GND_D1	Ground.		GND
D7	GND_D7	Ground.		GND
D8	GND_D8	Ground.		GND
D9	GND_D9	Ground.		GND
E2	GND_E2	Ground.		GND
E5	GND_E5	Ground.		GND
F2	GND_F2	Ground.		GND
F6	GND_F6	Ground.		GND
F9	GND_F9	Ground.		GND
G1	GND_G1	Ground.		GND
G2	GND_G2	Ground.		GND
G7	GND_G7	Ground.		GND
H10	GND_H10	Ground.		GND
H3	GND_H3	Ground.		GND
H8	GND_H8	Ground.		GND
J8	GND_J8	Ground.		GND
J9	GND_J9	Ground.		GND
K2	GND_K2	Ground.		GND
K4	GND_K4	Ground.		GND
K5	GND_K5	Ground.		GND
K8	GND_K8	Ground.		GND
L1	GND_L1	Ground.		GND
L5	GND_L5	Ground.		GND
L7	GND_L7	Ground.		GND
N3	GND_N3	Ground.		GND
P3	GND_P3	Ground.		GND
P6	GND_P6	Ground.		GND
P8	GND_P8	Ground.		GND
R9	GND_R9	Ground.		GND

K6	P0_2	UART RXD	VDD_33	I
J7	P0_3	UART TXD	VDD_33	O
J6	P0_5	EN pin	VDD_33	I
M2	P1_0	INT pin	VDD_33	O
M3	P1_1	MSG pin	VDD_33	I
R5	P11_2	QSPI_CS	VDDD	I/O
N6	P11_3	QSPI_IO3	VDDD	I/O
M4	P11_4	QSPI_IO2	VDDD	I/O
P4	P11_5	QSPI_IO1	VDDD	I/O
P5	P11_6	QSPI_IO0	VDDD	I/O
M5	P11_7	QSPI_SCK	VDDD	I/O
M1	VBACKUP	VBACKUP is the supply to the backup domain. The backup domain includes the 32-kHz WCO, RTC, and backup registers. It can generate a wake-up interrupt to the chip via the RTC timers or an external input. It can also generate an output to wakeup external circuitry. It is connected to VDDD when not used as a separate battery backup domain. VBACKUP provides the supply for Port 0. Min. is 1.4 V in Backup Mode		PWR
K1	VDD_NS	Power Supply for PSoC 62 Buck regulator	VDDD	PWR
H2	VDD_USB	Power Supply for PSoC 62 USB	3.3V	PWR
R8	VDDA	Power Supply for PSoC 62 P9,P10 (analog peripherals)	1.7~3.6V	PWR
R3	VDDD	Power Supply for PSoC 62 P1,P5,P6,P7,P8,P11,P12,XRES	1.7~3.6V	PWR
H1	VDDIO_WL	Power Supply for CYW43438 Digital I/O. Connect it to VDDIOR.	VDDIOR	PWR
J1	VDDIOR	Power Supply for PSoC 62 P2, P3, P4. Connect it to VDDIO_WL	1.8V	PWR
F1	WLAN_VBAT	Main Power Supply for CYW43438	3.2~4.8V	PWR
E1	WLAN_VBAT	Main Power Supply for CYW43438	3.2~4.8V	PWR
J2	XRES_L	External reset I/O pin(pulled up by a 4.7K ohms resistor internally)	VDDD	I

1.5 Layout Guide and SMT Process Notification

For correctly designing AW-CU427-P in your device, you may need to refer to Layout Guide or SMT Process Notification, please contact with AzureWave Technical Support Portal

2. AWS Command Example

Please find the command details in **AWS CONNECTOR AT Command Set**.

Below are commands for the demo:

1. Turn on Wi-Fi module: **AT+WIFI_On**
2. Retrieve the Wi-Fi AP | Station Mode: **AT+WIFI_GetMode**
3. Perform a Wi-Fi network scan: **AT+WIFI_Scan**
4. Set and store the Wi-Fi AP information when AW-CU427-P in Station Mode:

   AT+WIFI_SetAP=**SSID**,**password,security type**
   SSID: SSID of AP (case sensitive)
   password: password for AP (case sensitive)
   security type: OPEN | WEP | WPA | WPA2

5. Connect to the AP: **AT+WIFI_Connect**
6. Define and store Thing-specific configuration:

   AT+THING_Set=client ID,endpoint,client certificate,client private key
   client ID: Thing name(Client ID)
   endpoint: AWS IoT endpoint URL
   client certificate: Certificate for this Thing
   client private key: Private key for this Thing

7. Connect the client to MQTT broker: **AT+MQTT_Connect**
8. Subscribe to and save MQTT topic: **AT+MQTT_Subscribe**=<topic>,<qos>
9. Publish to MQTT topic:

   AT+MQTT_Publish=topic,message,qos
   topic: Topic to publish to
   message: Message to publish
   qos: 0 | 1

2.1 Getting Started with AWS IoT Core

Step 0**:** The below link is a documents of how to setup AWS IOT, you can refer to it for full AWS IOT knowledge.

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

But, if you want to setup AzureWave AWS Connector, you would just refer to the following steps.

Step 1: Create AWS Account, Create an IAM user.

Please refer to the below link to setup AWS Account and IAM user.

https://docs.aws.amazon.com/iot/latest/developerguide/setting-up.html

If you have created an IAM user, please refer to the following setting to connect these two policies (AmazonFreeRTOSFullAccess, AWSIoTFullAccess) to your IAM.

https://docs.aws.amazon.com/freertos/latest/userguide/freertos-account-and-permissions.html

Step 2: Create a thing.

A thing represents a specific device or instance that can communicate with AWS IOT.

Please refer to the following link to create a thing.

https://docs.aws.amazon.com/iot/latest/developerguide/create-aws-thing.html

Step 3: Register a device

This step will create certificate and private key. You can use certificate, private key, thing name and endpoint as **AT+THING_Set** command parameter. After this command executing, the four parameter will be provision to our connector. After provisioning, you can connect to AWS IOT with MQTT or SHADOW operation.

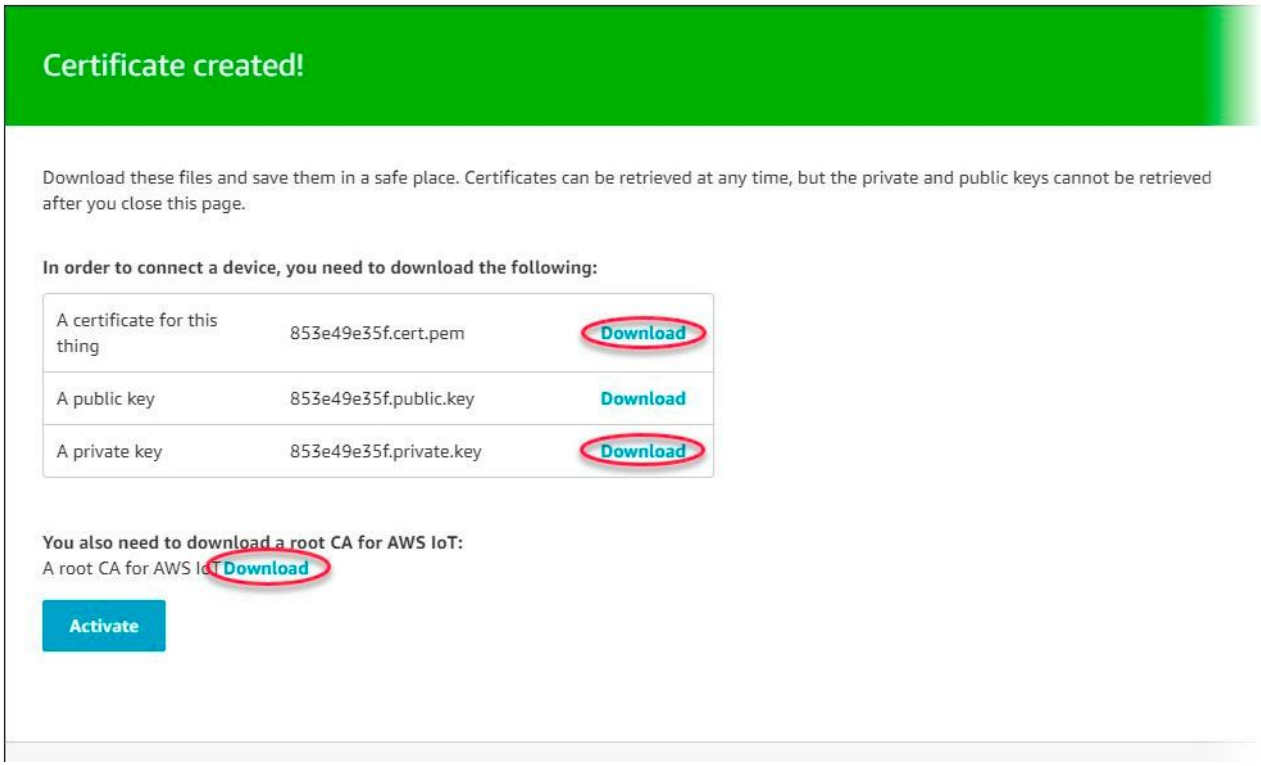
Please refer to the steps at the following link.

https://docs.aws.amazon.com/iot/latest/developerguide/register-device.html

After finishing the steps, please notice the following two actions:

- Download certificate and private key

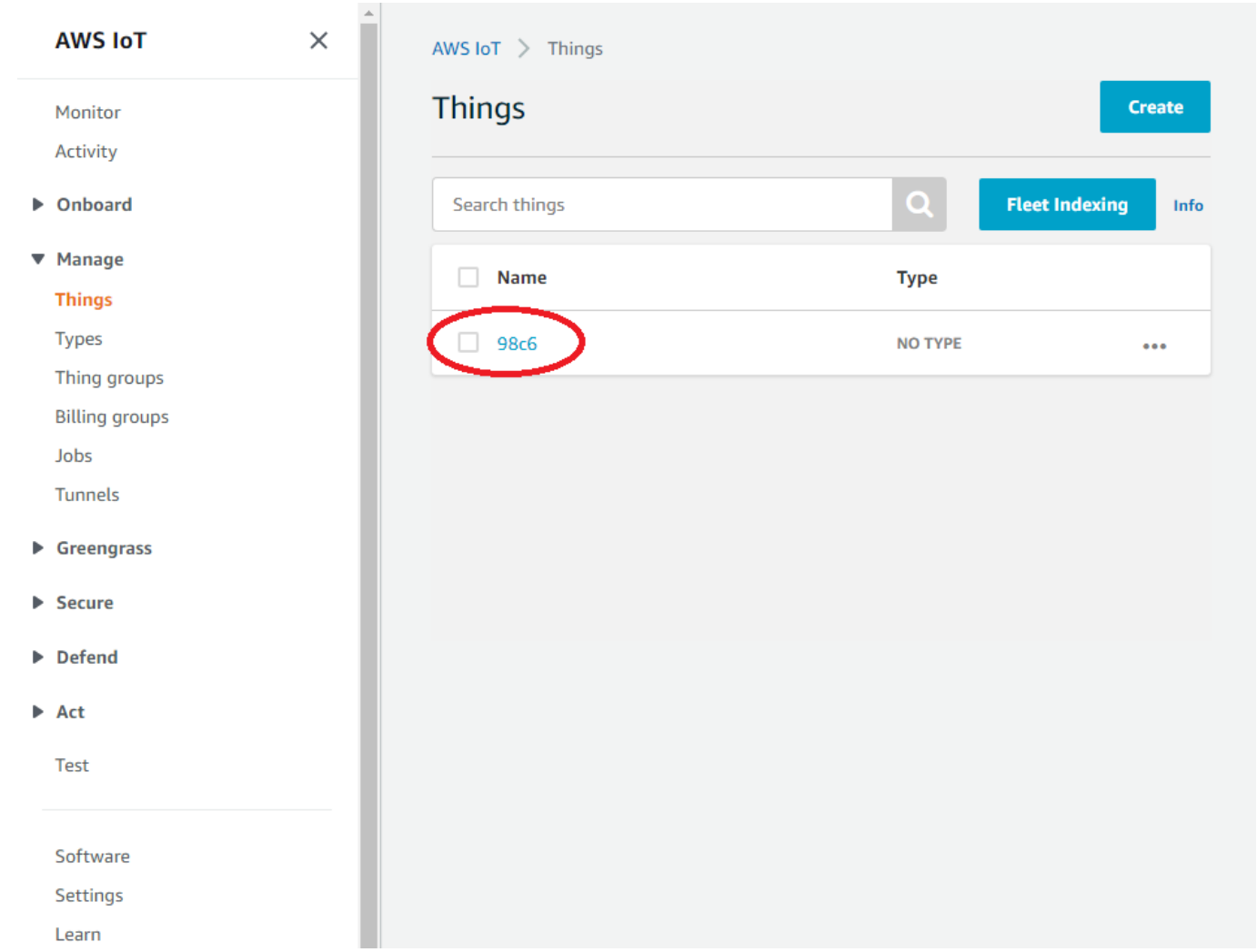
In Create and activate a device certificate chapter, please download and keep the certificate and private key. Because they will be used when sending the **AT+THING_Set** command.

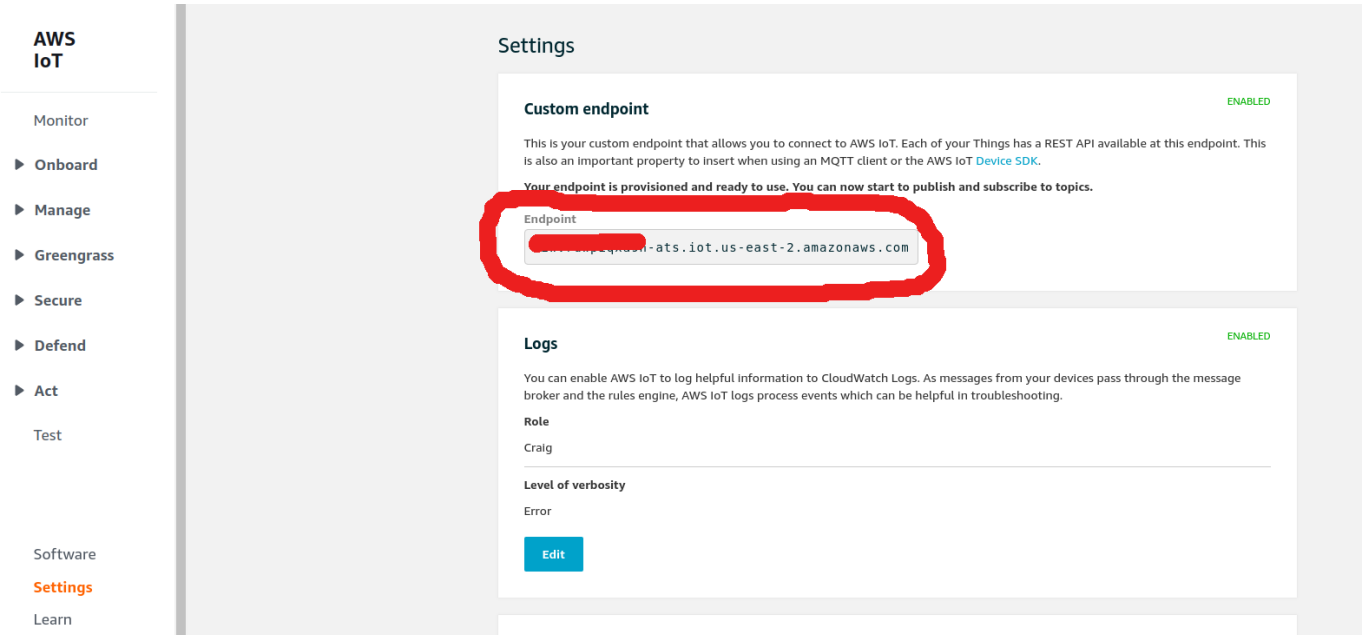


- Thing Name and Endpoint

These two data will also be used for **AT+THING_Set** command.

You can find out thing name in Manage > Things submenu, and endpoint in settings of AWS IOT Console at console.aws.amazon/iot.





2.2 Publish and Monitor MQTT message on the cloud

Step 0: Make sure the AP under test is connected to the internet using other Wi-Fi enabled devices. Assume the SSID, password and security type of the AP is MySSID, MyPassword, wpa2.

Step 1: Connect the AW-CU427-P to the system and turn off the wireless devices near the AW-CU427-P (except for the device under test). Turn on the Wi-Fi module of the AW-CU427-P using **AT+WIFI_On**

Step 2: Check if AW-CU427-P is in station mode using **AT+WIFI_GetMode**

Step 3: Set and store information of the AP using **AT+WIFI_SetAP= MySSID,MyPassword,wpa2**

AT+WIFI_SetAP=**SSID**,**password,security type**
SSID: SSID of AP (case sensitive)
password: password for AP (case sensitive)
security type: OPEN | WEP | WPA | WPA2

Step 4: Connect to the AP: **AT+WIFI_Connect**

Step 5**:** Define and store Thing-specific configuration using **AT+THING_Set** command

AT+THING_Set=client ID,endpoint,client certificate,client private key
client ID: Thing name(Client ID)
endpoint: AWS IoT endpoint URL
client certificate: Certificate for this Thing (downloaded in 3.1 step 3)
client private key: Private key for this Thing (downloaded in 3.1 step 3)

You should create command as format below:

AT+THING_Set=98c6, a3qjEXAMPLEffp-ats.iot.ap-northeast-1.amazonaws.com, -----BEGIN CERTIFICATE-----\n…base64 data…\n-----END CERTIFICATE-----\n, -----BEGIN RSA PRIVATE KEY-----\n…base64 data…\n-----END RSA PRIVATE KEY-----\n

Note: a3qjEXAMPLEffp is just an example endpoint, your endpoint URL should replace it. Please follow the instructions to find the endpoint.

- Navigate to the AWS IoT console
- Choose Settings in the navigation pane
- The endpoint is can be found under Custom endpoint

Step 6: Connect the client to MQTT broker: **AT+MQTT_Connect**.

Step 7: Subscribe to and save MQTT topic using **AT+MQTT_Subscribe=iotdemo/1,0**

Step 8: Publish to MQTT topic using **AT+MQTT_Publish=iotdemo/1, hello, 0**

AT+MQTT_Publish=topic,message,qos
topic: Topic to publish to
message: Message to publish
qos: 0 | 1

Step 9: Use the MQTT client in the AWS IoT console to monitor the messages that device sends to the AWS Cloud.

Sign in to the AWS IoT console.

https://console.aws.amazon.com/iotv2/

In the navigation pane, choose **Test** to open the MQTT client.

In **Subscription topic**, enter **iotdemo/#**, and then choose **Subscribe to topic**.

You should see the message send from device as like below.

2.3 Over-the-Air Updates

This section descripts how to setup OTA on AWS server side and how to create an OTA job for our AWS connector to update firmware.

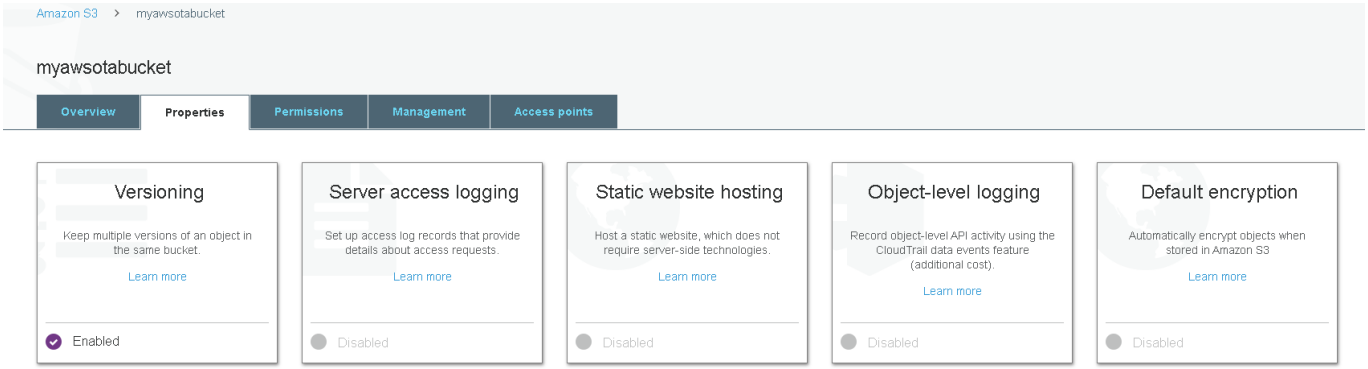
The first three steps descript what requirements of OTA on AWS server side you have to setup. The last step descripts how to create an OTA job.

Create S3 bucket to store update.

AWS S3 is an object storage service and we will put update firmware in it.

Please reference https://docs.aws.amazon.com/freertos/latest/userguide/dg-ota-bucket.html

Note: Please make sure you have enable versioning on you S3 bucket setting.

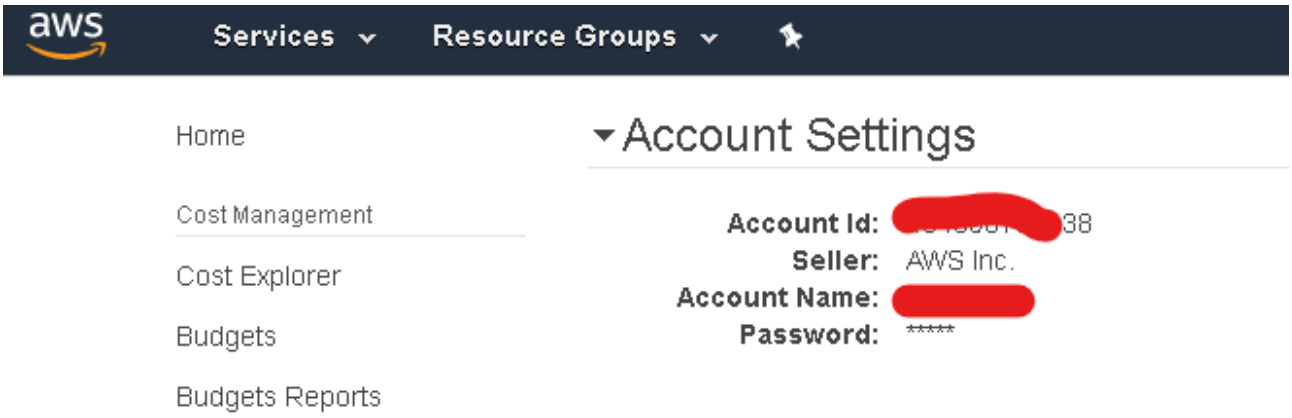


Create Service Role and Policy.

This section will descript how to setup Role and Policy to have right to access AWS S3 storage.

- Service Role
  Please reference
  https://docs.aws.amazon.com/freertos/latest/userguide/create-service-role.html

  If you want to find out your Account id, please enter the following link.
  https://console.aws.amazon.com/billing/home?#/account



- User Policy
  Please reference
  https://docs.aws.amazon.com/freertos/latest/userguide/create-ota-user-policy.html

Create code-signing certificate

Because there are several configures in this topic, please use the following link to complete code-signing certificate.

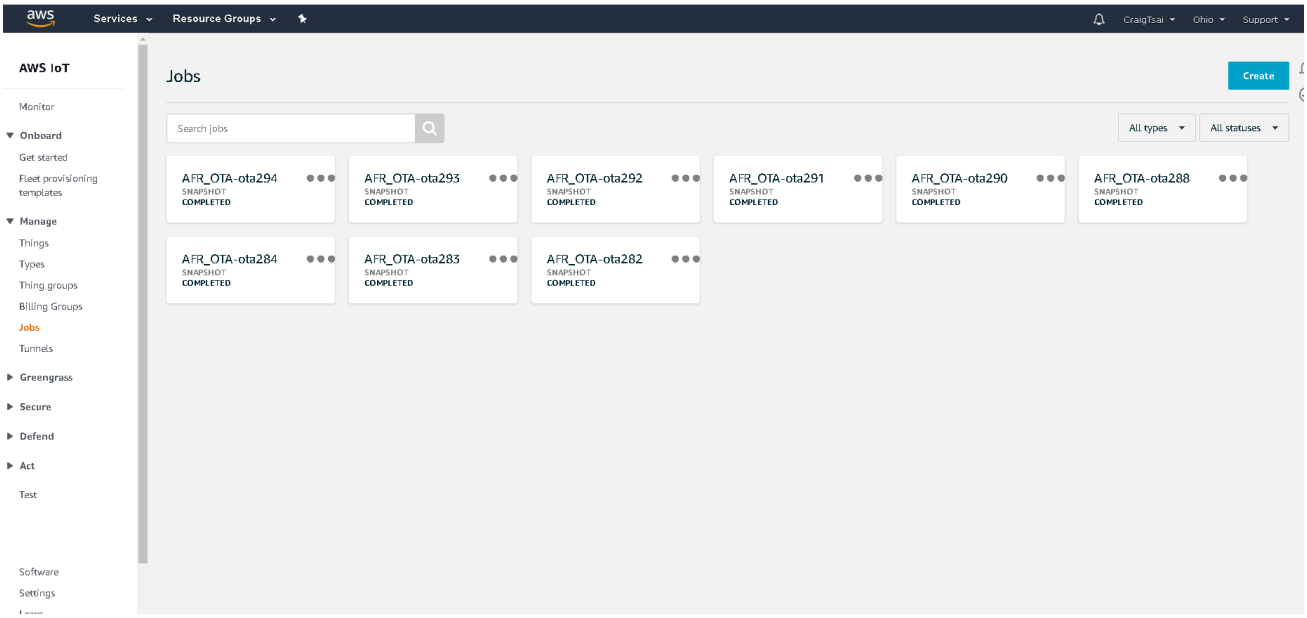
https://docs.aws.amazon.com/freertos/latest/userguide/ota-code-sign-cert-win.html

AWS CLI and openssl are necessary softwares to complete this task.

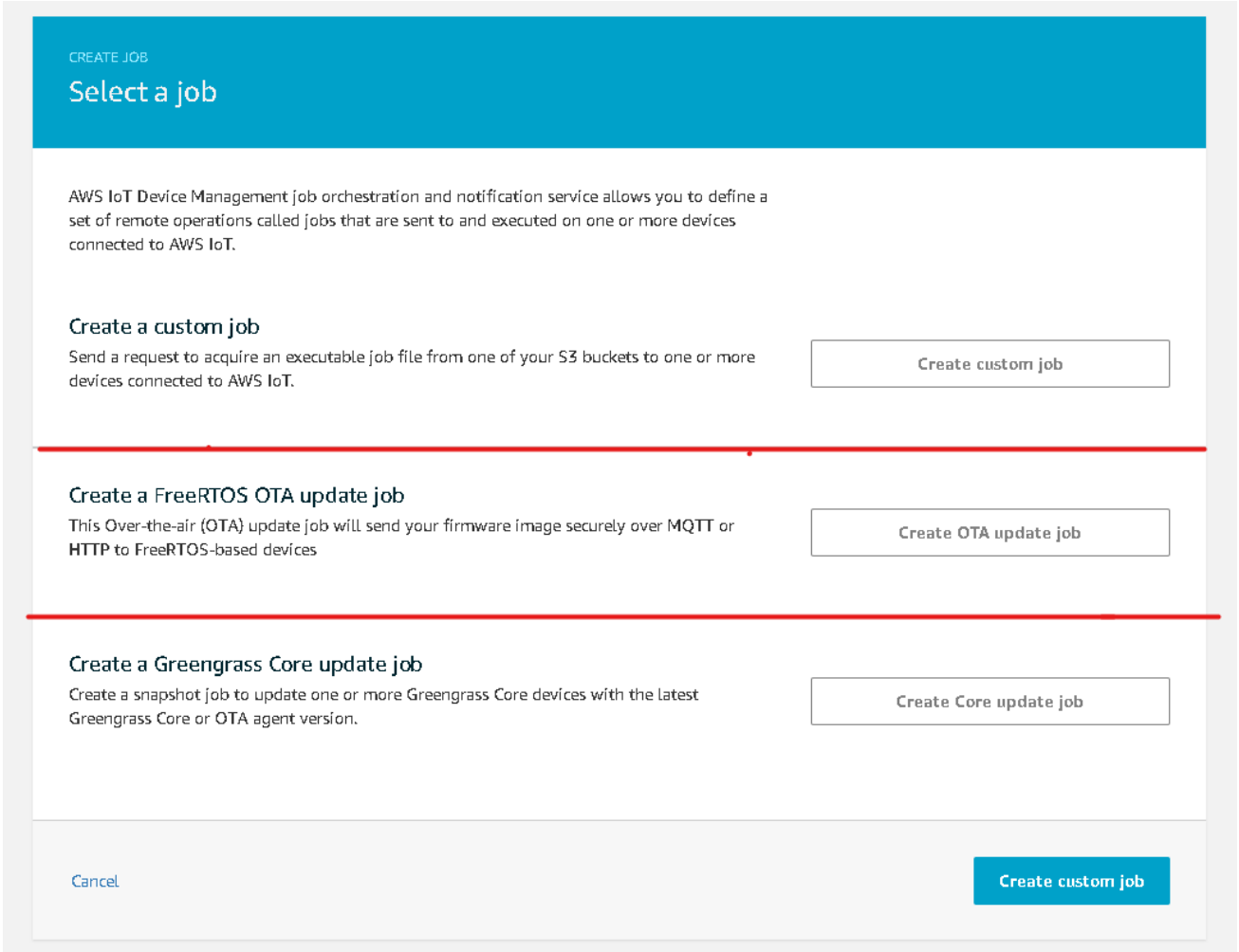
Create OTA update Job

After all prepared procedure finished, you can create an OTA update job on this thing name and use AT+MOD_OTA command to trigger OTA update procedure.

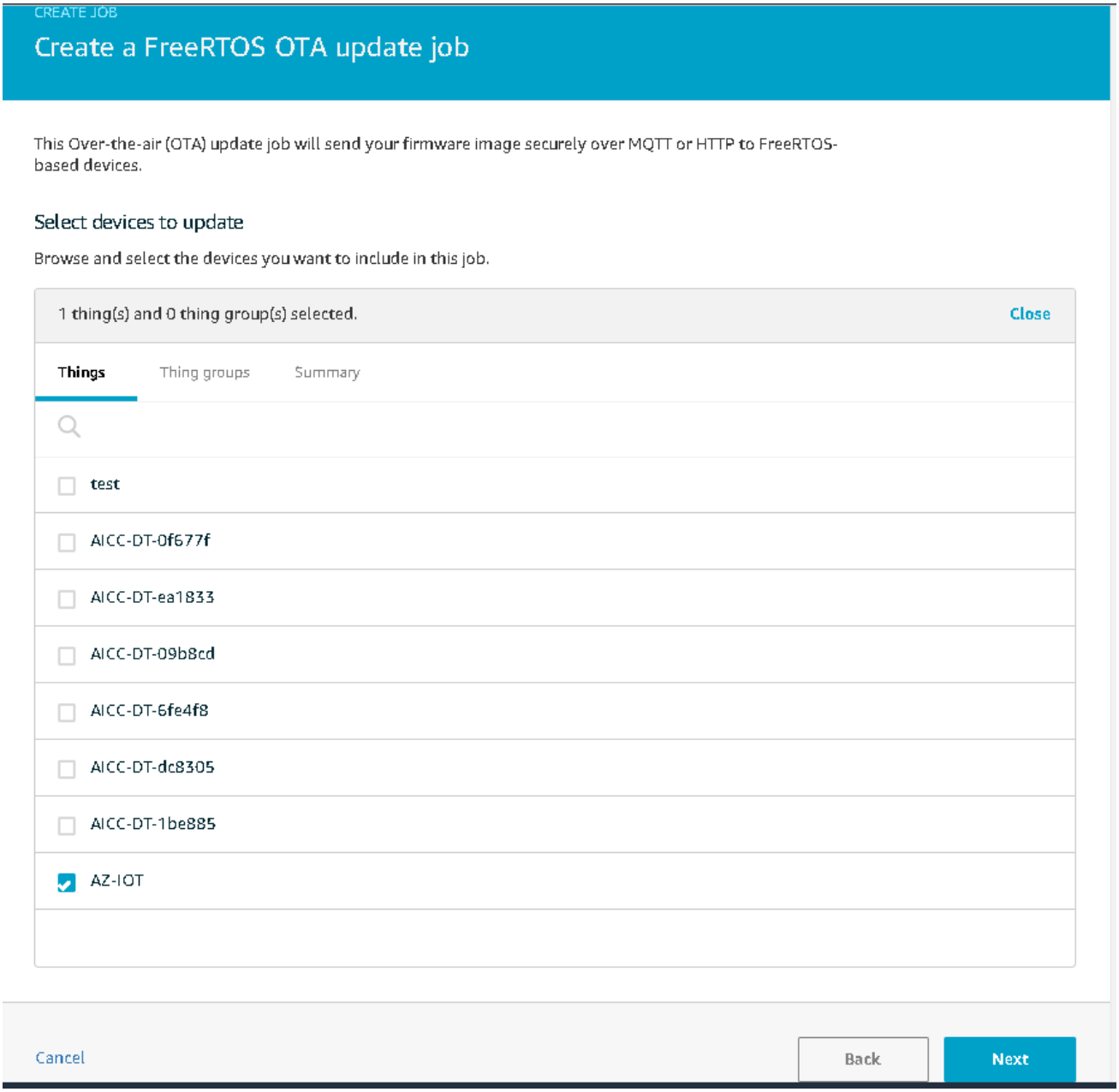
1. Create OTA Job
   Please enter AWS IOT console, choose Manager, and press Jobs and then press Create button.



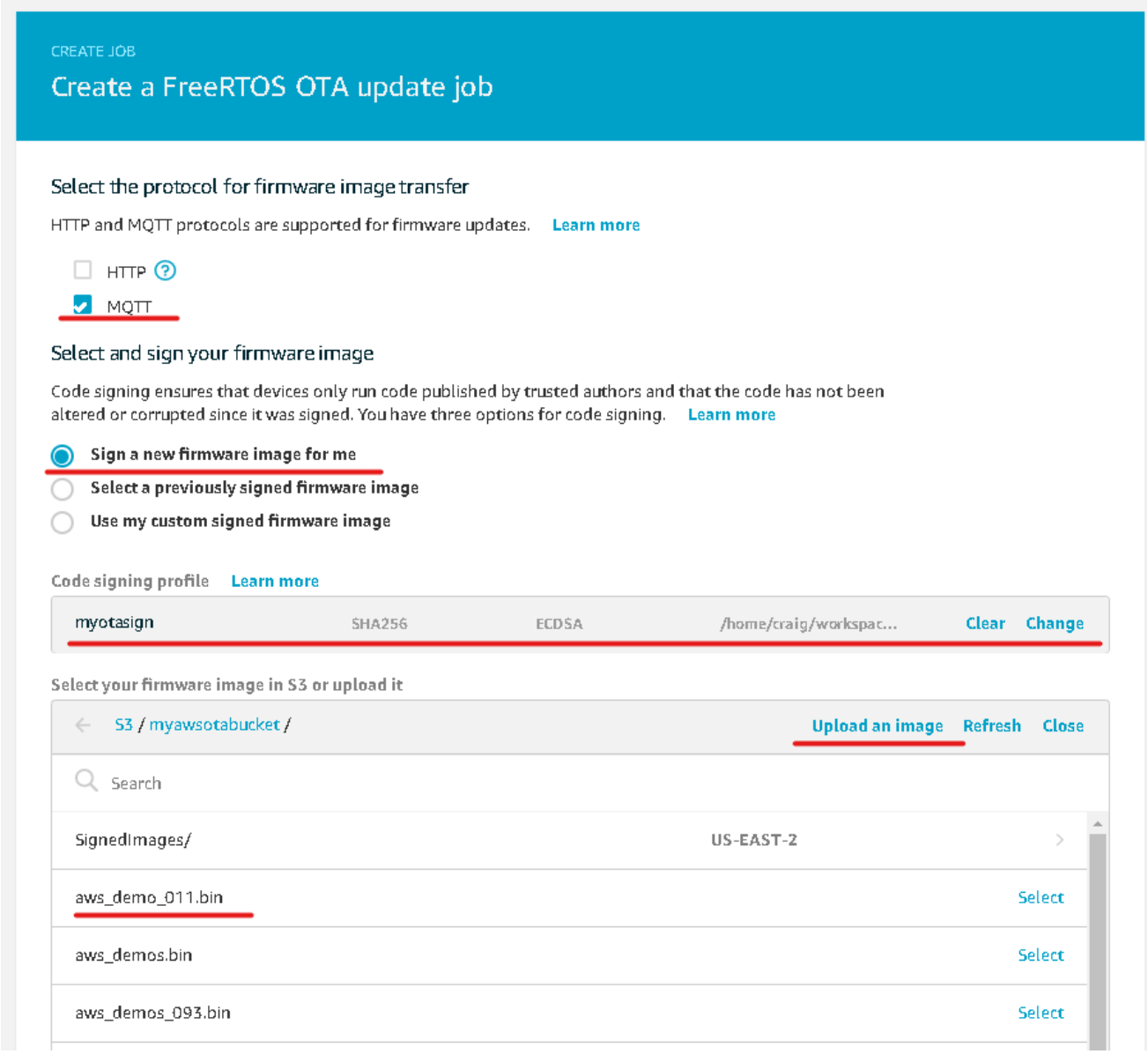
2. Please choice a FreeRTOS OTA update job and press “**create OTA update job**”



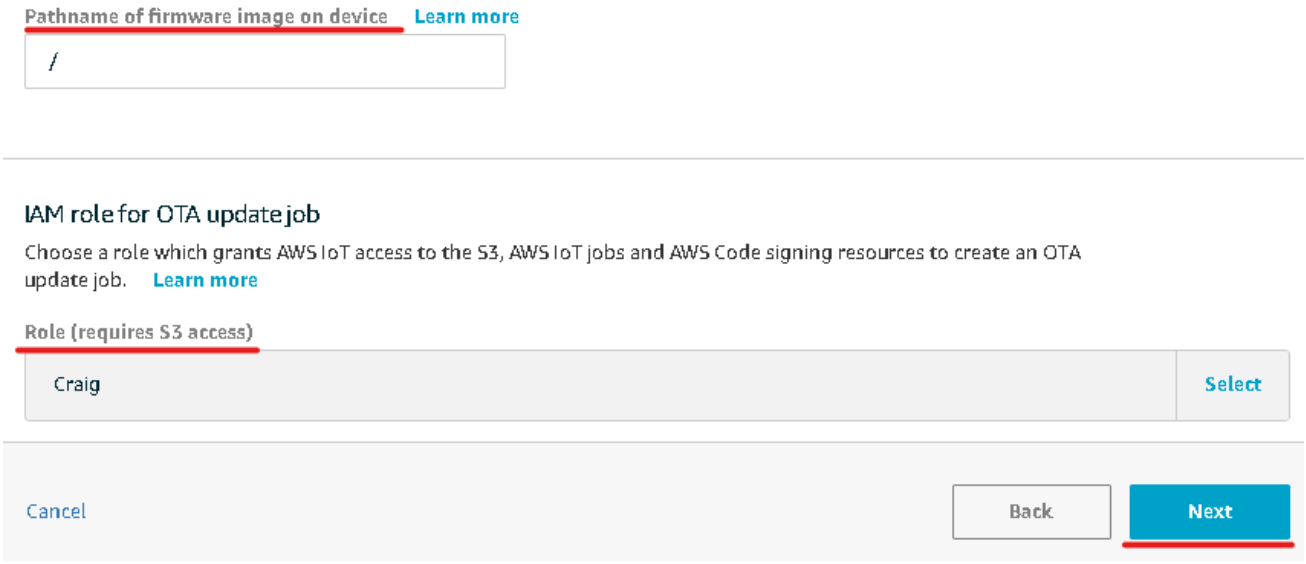
3. Select a Things to create OTA job and then press NEXT



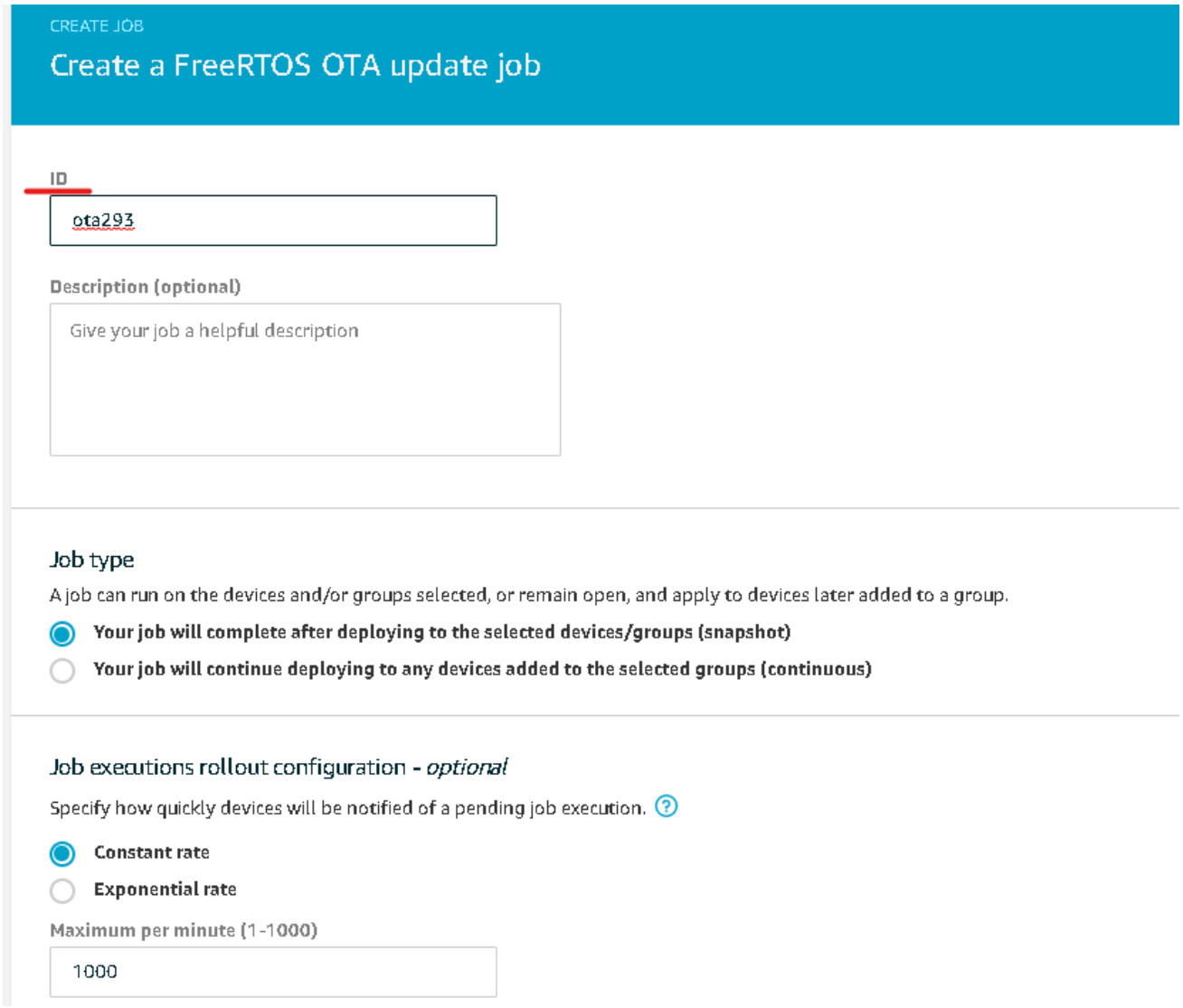
4. On next page, please only select **MQTT** to be update protocol and select **Sign a new firmware for me**. Then choose the **code signing profile** we create on previous chapter. After that, you can select a exist firmware or upload a new image from local.



Then, please enter slash in **Pathname of firmware image on device** and choose the **Role** we create on previous chapter. Then press NEXT button.



5. Entering a **unique ID** in ID field. You can't use the same ID you created before. Other fields just keep default setting and press Create button.



6. Finally, you can see a job to be created on Jobs page. If you don't see it, please press f5 to refresh the webpage. The Job will be in IN PROGRESS status.