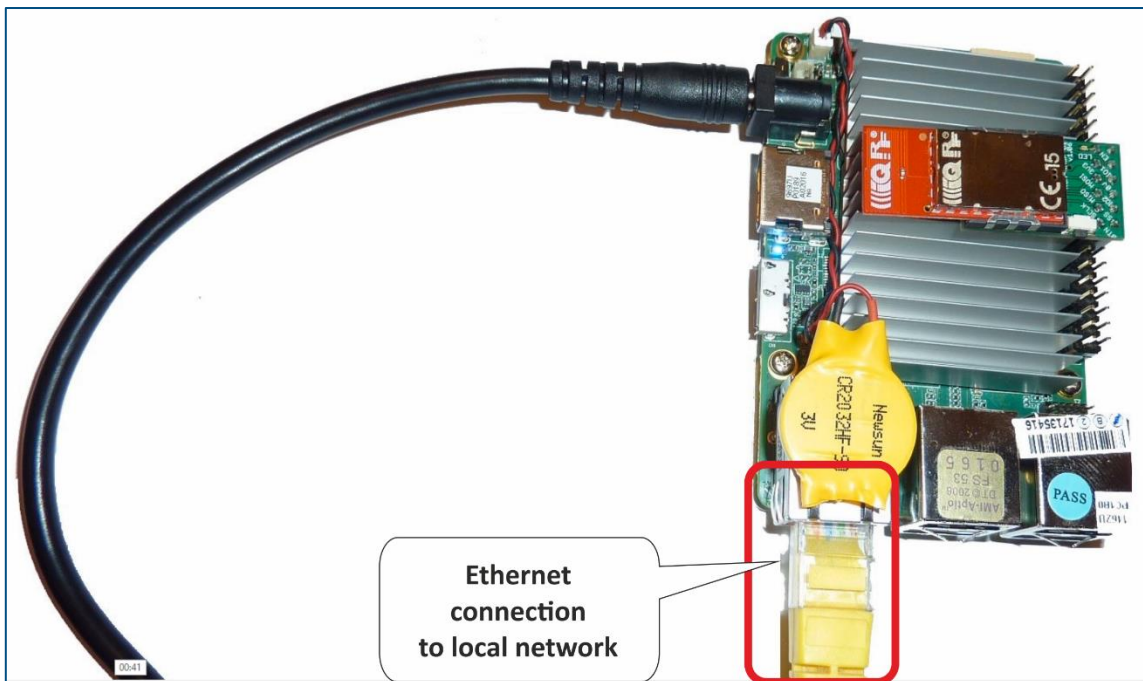


IoT Starter Kit – Part 3: Connect to the cloud – AWS IoT

IoT Starter Kit is designed in the way to be connectable to different clouds via bidirectional MQTT channel. So, you can collect, store, process and visualize data in a cloud or you can send your commands to the IQRF network remotely. In this part, we will configure the UP board to communicate with the Amazon Web Services (AWS) through the MQTT channel.

1 Local network

Connect your UP board to your local network so it can obtain an IP address using DHCP. In the following steps, you will enter this address into your web browser on your computer (which is in the same local network as the UP board) and configure your gateway through the IQRF Daemon Web application.



2 Amazon Web Services account

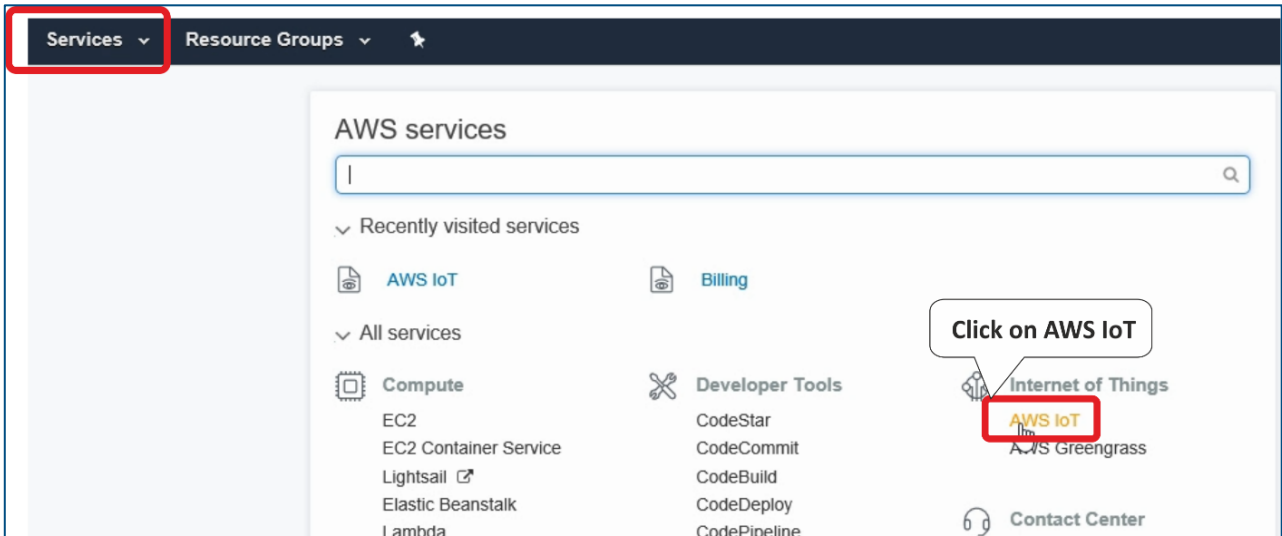
First, create an Amazon Web Services account (aws.amazon.com). You must fill in your personal or company data and add your credit card details. Your credit card will be used for payments in a case you exceed limits of the selected services.

The image shows a screenshot of the AWS website. At the top, there is a navigation bar with links for Pricing, Getting Started, Documentation, Software, Support, Customers, Partners, and More. On the right side of the navigation bar, there are links for English, My Account, and a highlighted 'Create an AWS Account' button. Below the navigation bar, the main content area features the heading 'Start Building on AWS Today' and a sub-heading 'Create your AWS Account'. A callout box points to the 'Create an AWS Account' button. Below this, there is a form titled 'Create a new AWS Account' with fields for AWS account name, Email address, Password, and Confirm password. A 'Continue' button is located below the form. To the right of the form, there is a graphic of a laptop with a checkmark and the text 'AWS Accounts Include 12 Months of Free Tier Access'. Below this graphic, it says 'Including use of Amazon EC2, Amazon S3, and Amazon DynamoDB' and 'Visit aws.amazon.com/free for full offer terms'. A green arrow points from the 'Create an AWS Account' button to the sign-up form.

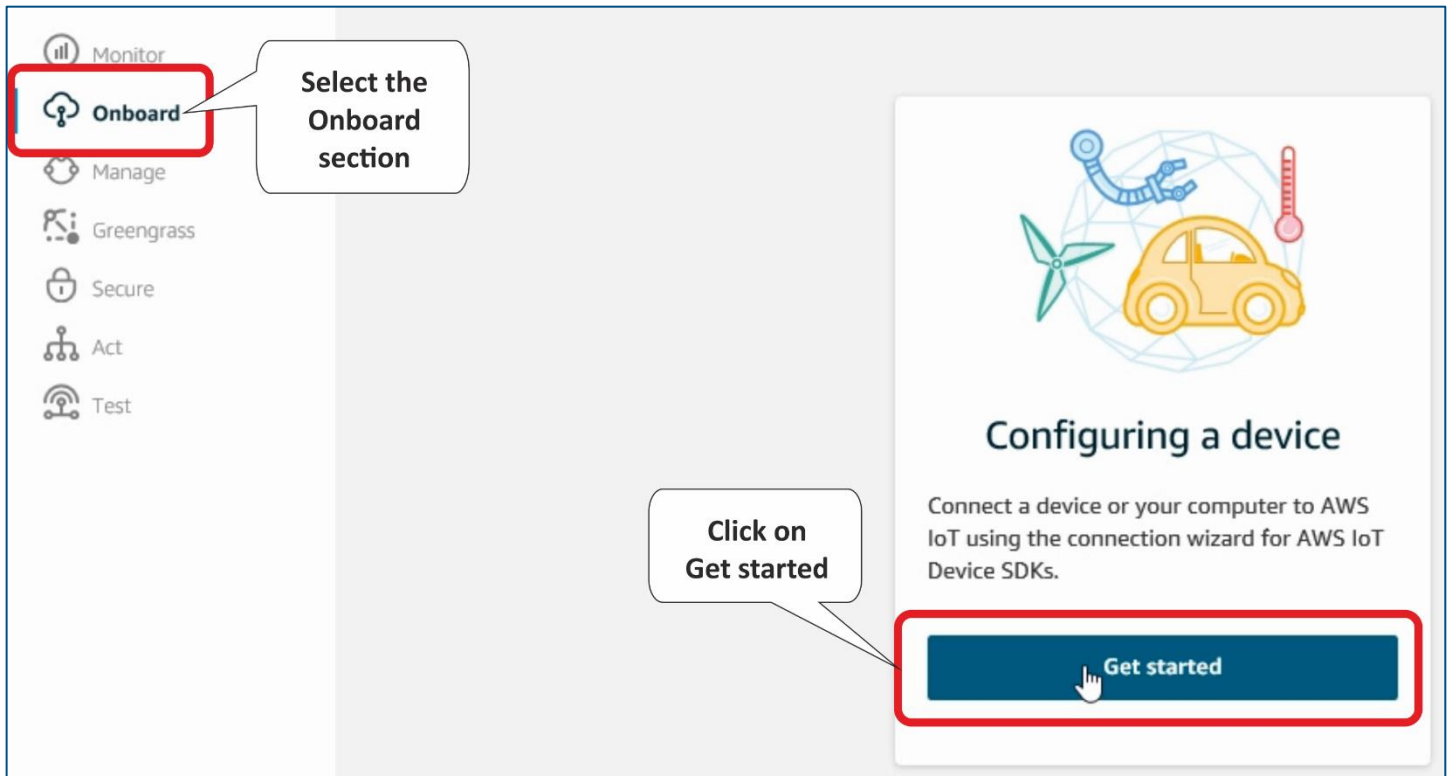
3 Set up the connection

To set up the connection between AWS and your UP board, you need to do some configuration steps on both sides.

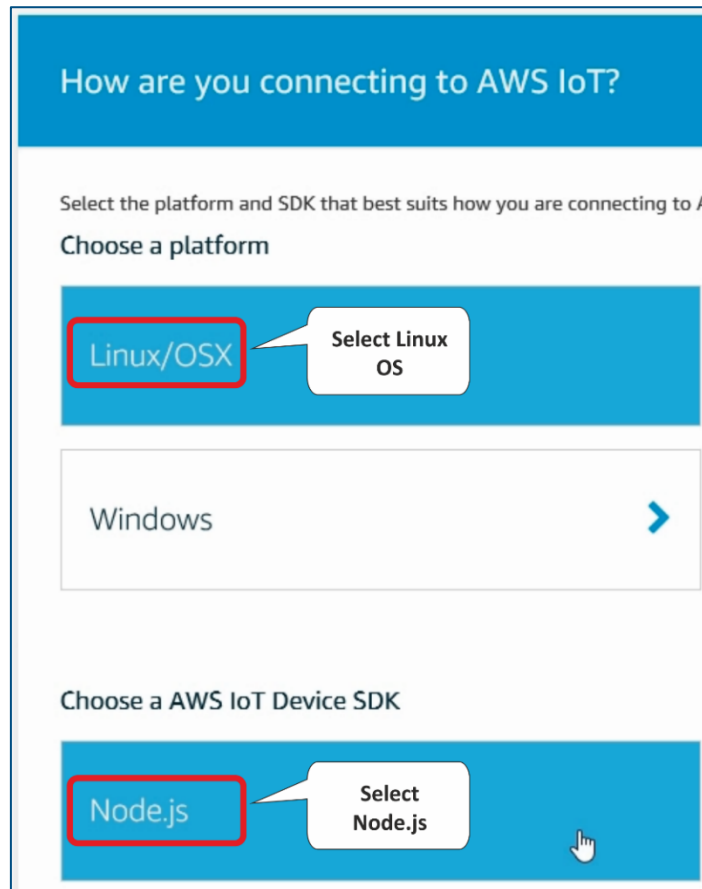
In **Services**, in the **Internet of Things** section of AWS, find **AWS IoT**.



Click on **Get started** in the **Onboard** section. You will register your device, download the connection kit, and configure and test the connection with your device.

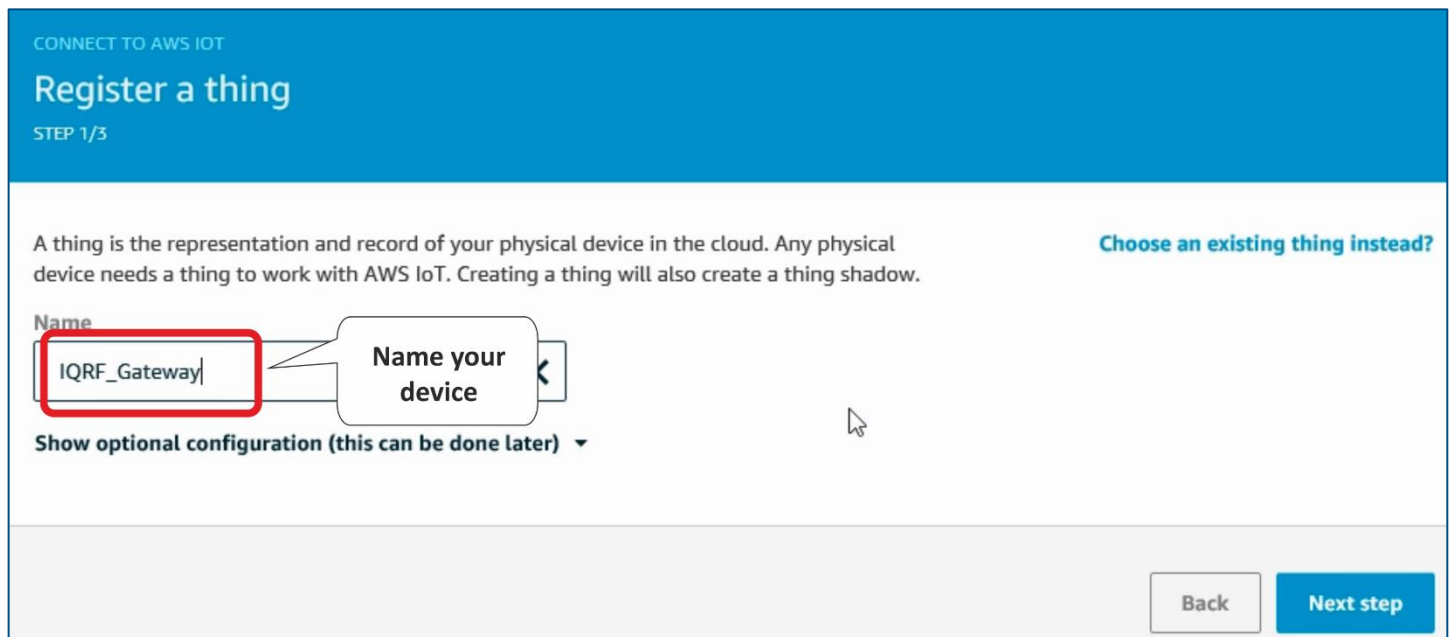


Set up how you will be connected to the AWS IoT. Select the **Linux** operating system and **Node.js** as the AWS IoT Device SDK.



Enter the name of your connected device.

Note: You can choose your own name. In that case in later steps, you need to use the given name.



Download the connection kit to get a certificate and keys for a secure MQTT connection.

CONNECT TO AWS IOT

Download a connection kit

STEP 2/3

The following AWS IoT resources will be created:

A thing in the AWS IoT registry	IQRF_Gateway
A policy to send and receive messages	IQRF_Gateway-Policy Preview policy

The connection kit contains:

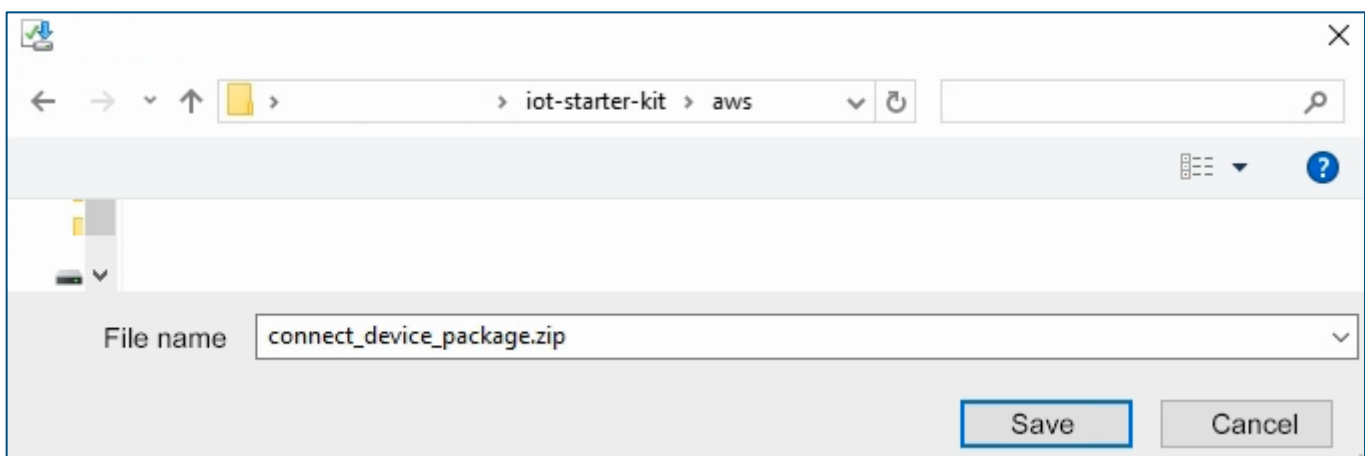
A certificate and private key	IQRF_Gateway.cert.pem, IQRF_Gateway.private.key
AWS IoT Device SDK	Node.js SDK
A script to send and receive messages	start.sh

Before your device can connect and publish messages, you will need to download the connection kit.

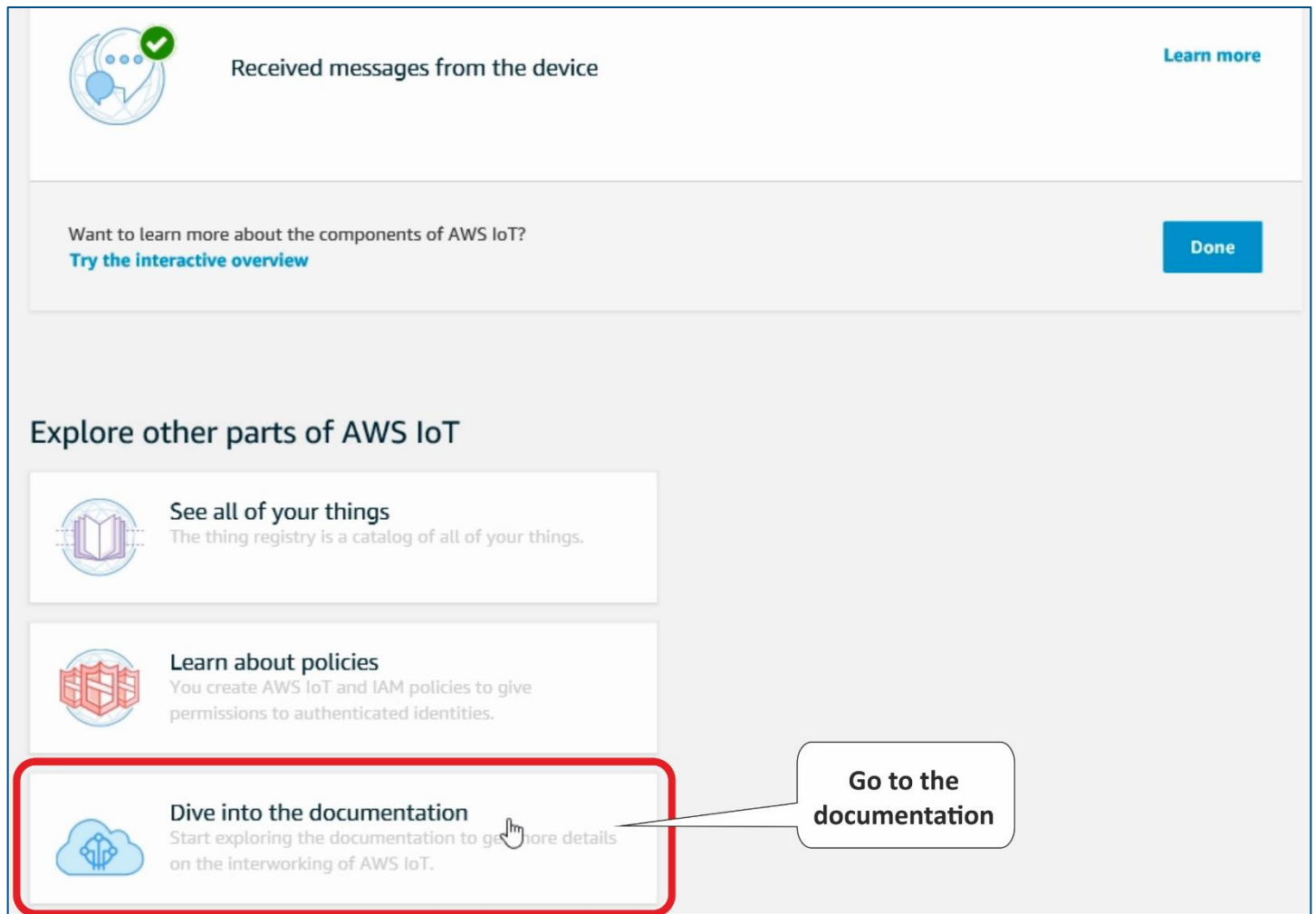
Download connection kit for

Linux/OSX Download connection kit

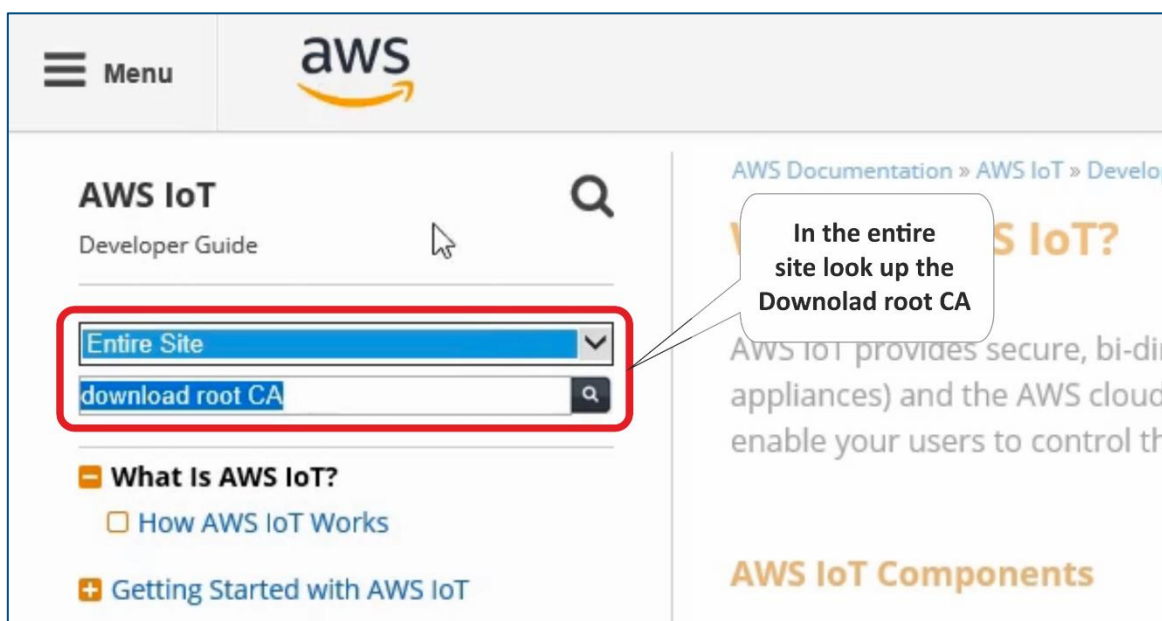
Save and unzip this file to your computer. Store the certificate and the keys for further use.



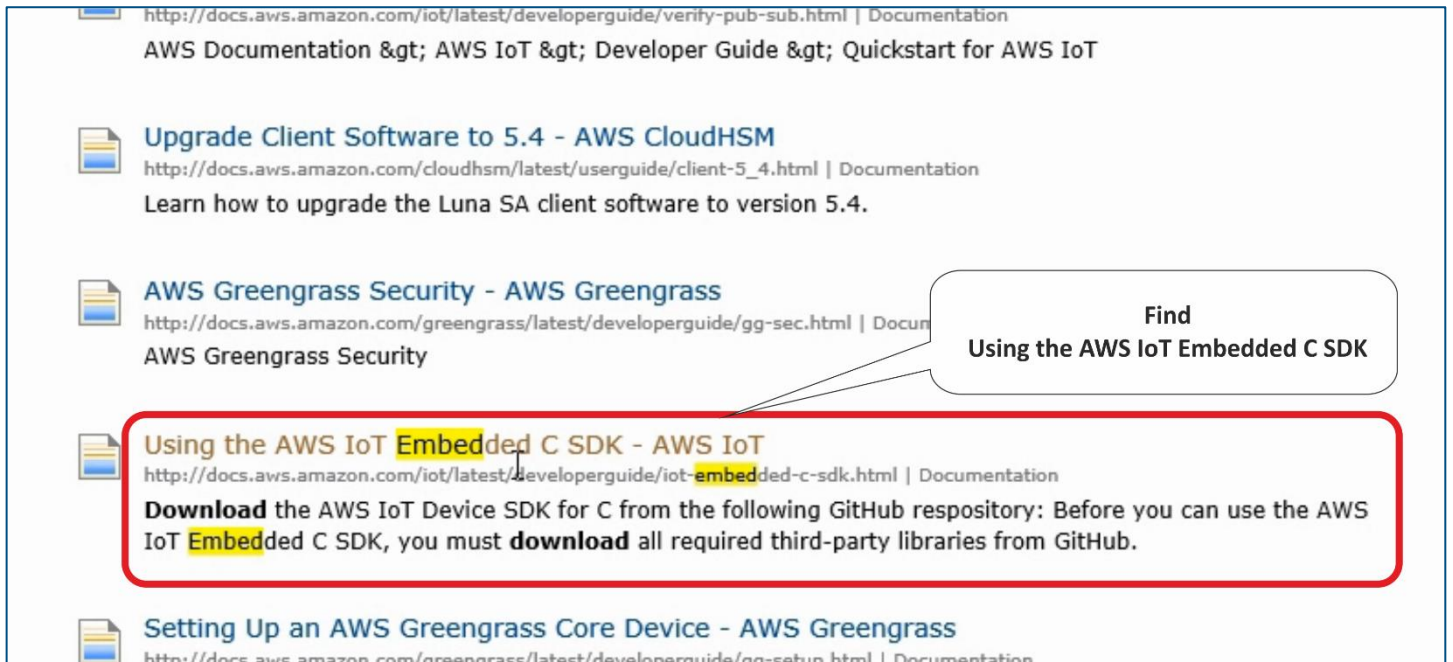
After saving process, go to the documentation.



Here, look up the **Download root CA** string. Search in the **Entire site** to be sure to find it.

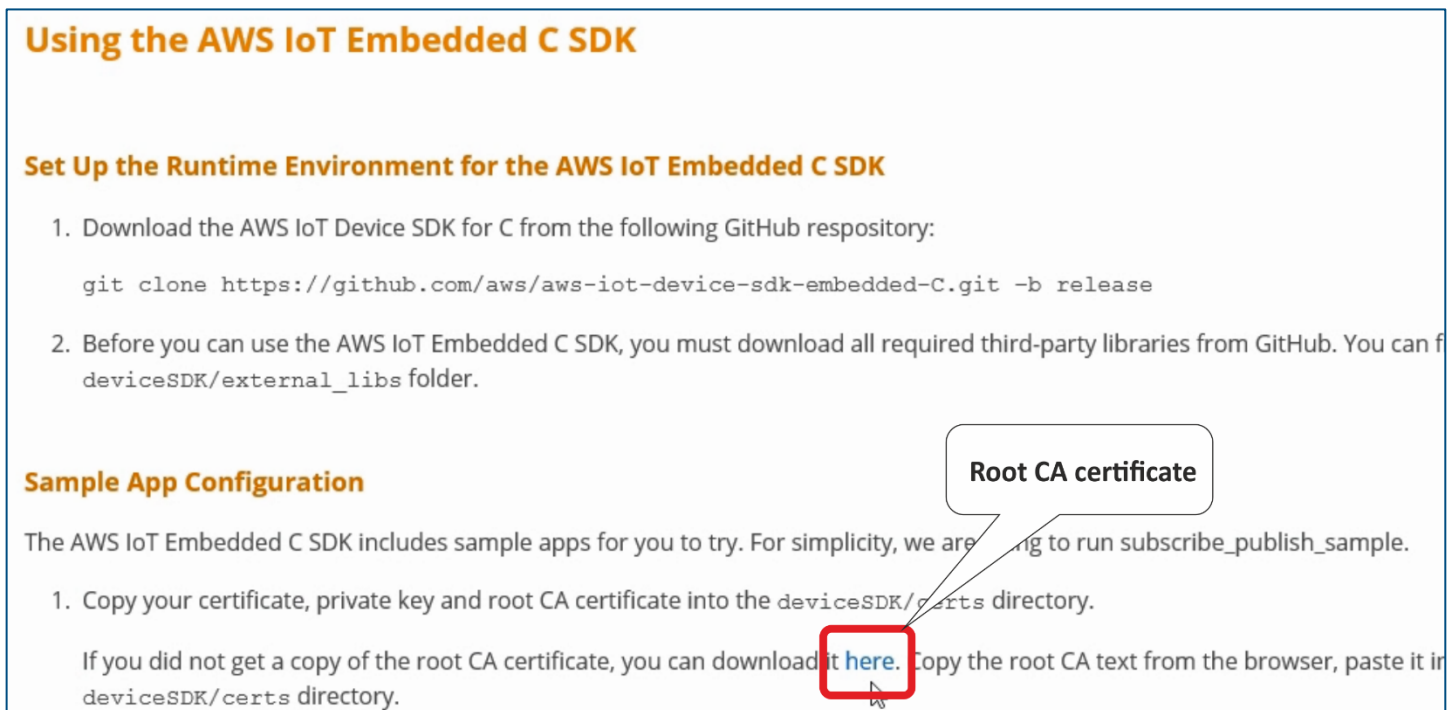


In the search results, find the article **Using the AWS IoT Embedded C SDK**. The number of records in a search result can exceed the page limit so you need to go through more pages.



The screenshot shows search results from AWS documentation. The first result is 'Upgrade Client Software to 5.4 - AWS CloudHSM'. The second is 'AWS Greengrass Security - AWS Greengrass'. The third result, 'Using the AWS IoT Embedded C SDK - AWS IoT', is highlighted with a red box. A callout bubble points to this result with the text 'Find Using the AWS IoT Embedded C SDK'. The highlighted result includes the URL 'http://docs.aws.amazon.com/iot/latest/developerguide/iot-embedded-c-sdk.html' and the text: 'Download the AWS IoT Device SDK for C from the following GitHub repository: Before you can use the AWS IoT Embedded C SDK, you must download all required third-party libraries from GitHub.'

Here you can find the root certificate.



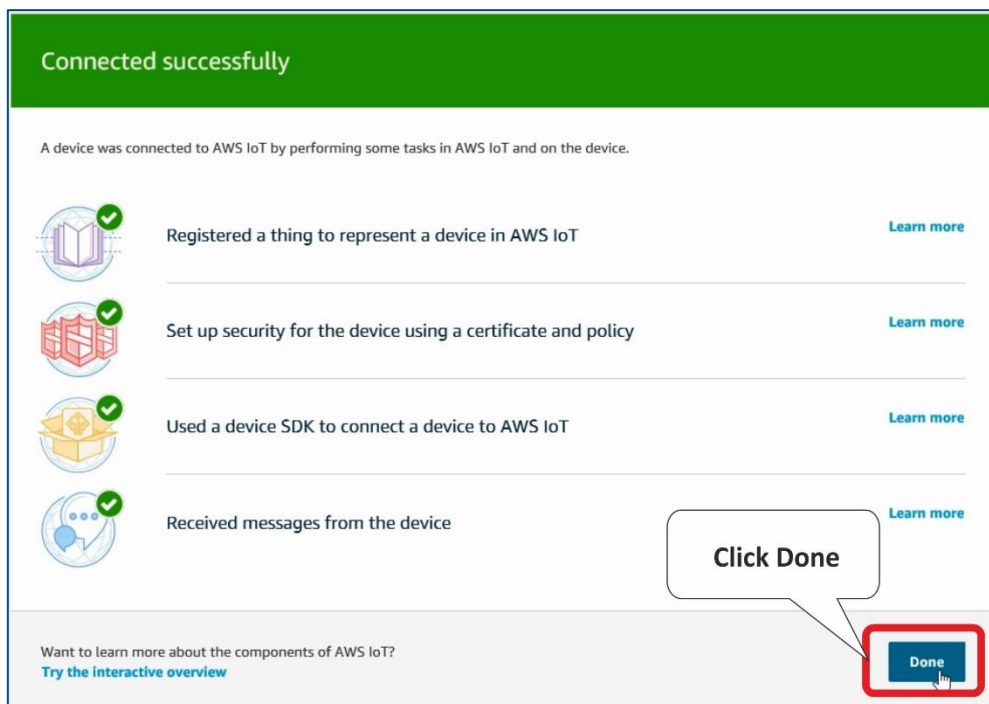
The screenshot shows the article 'Using the AWS IoT Embedded C SDK'. It has a sub-section 'Set Up the Runtime Environment for the AWS IoT Embedded C SDK' with two steps. Step 1 includes a code block: `git clone https://github.com/aws/aws-iot-device-sdk-embedded-C.git -b release`. Step 2 says to download third-party libraries from GitHub. Below this is the 'Sample App Configuration' section. It says 'The AWS IoT Embedded C SDK includes sample apps for you to try. For simplicity, we are going to run subscribe_publish_sample.' Step 1 says to copy certificate, private key, and root CA certificate into the `deviceSDK/certs` directory. A callout bubble points to the text 'Root CA certificate' and a red box highlights the link 'it here' in the sentence 'If you did not get a copy of the root CA certificate, you can download it here. Copy the root CA text from the browser, paste it in deviceSDK/certs directory.'

Copy the string to a text file and save it as **rootCA.pem** to the directory with other certificates on your computer.

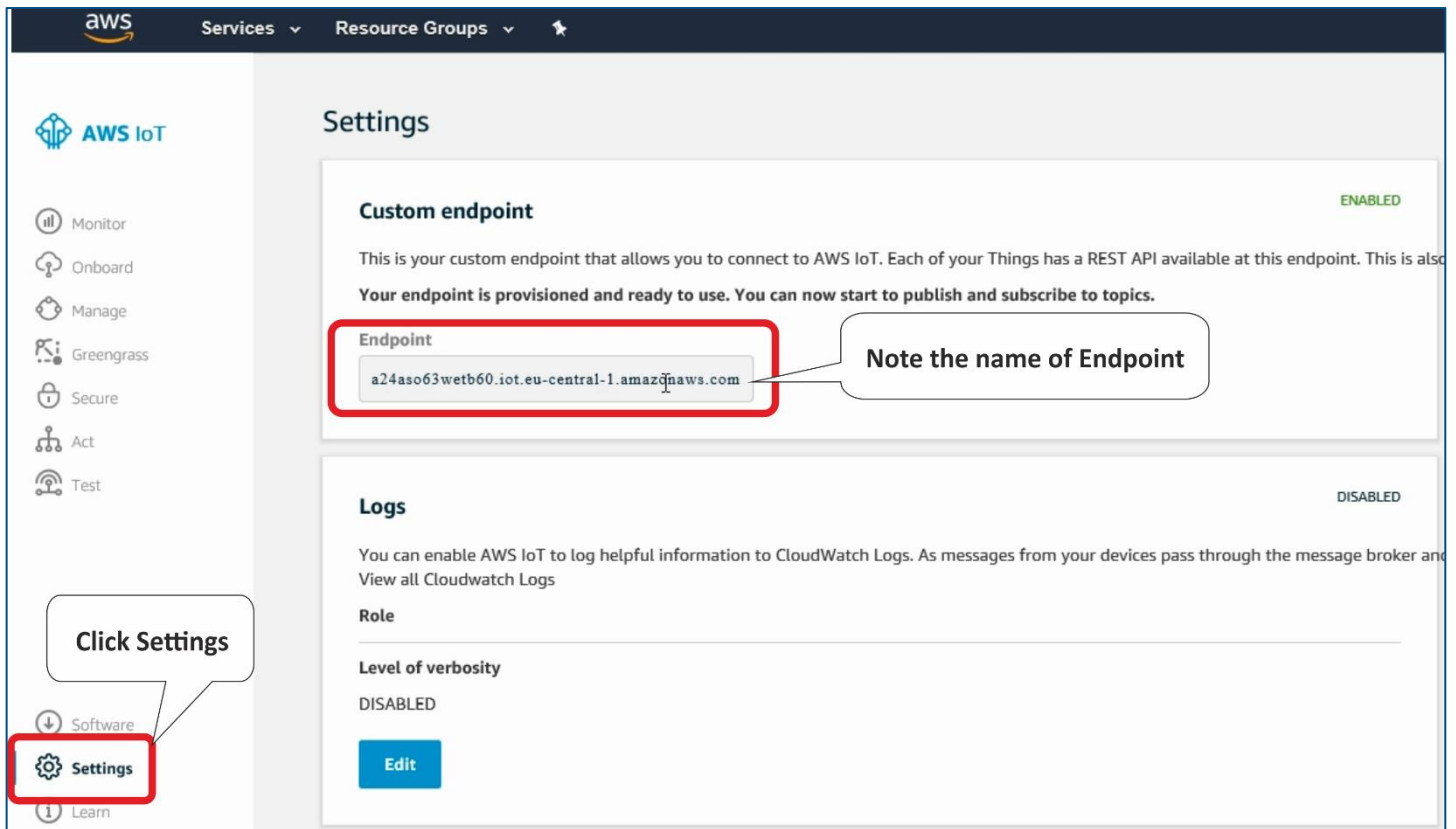
Note: You can choose your own name. In that case in later steps, you need to use the given name.



Next, click Done.

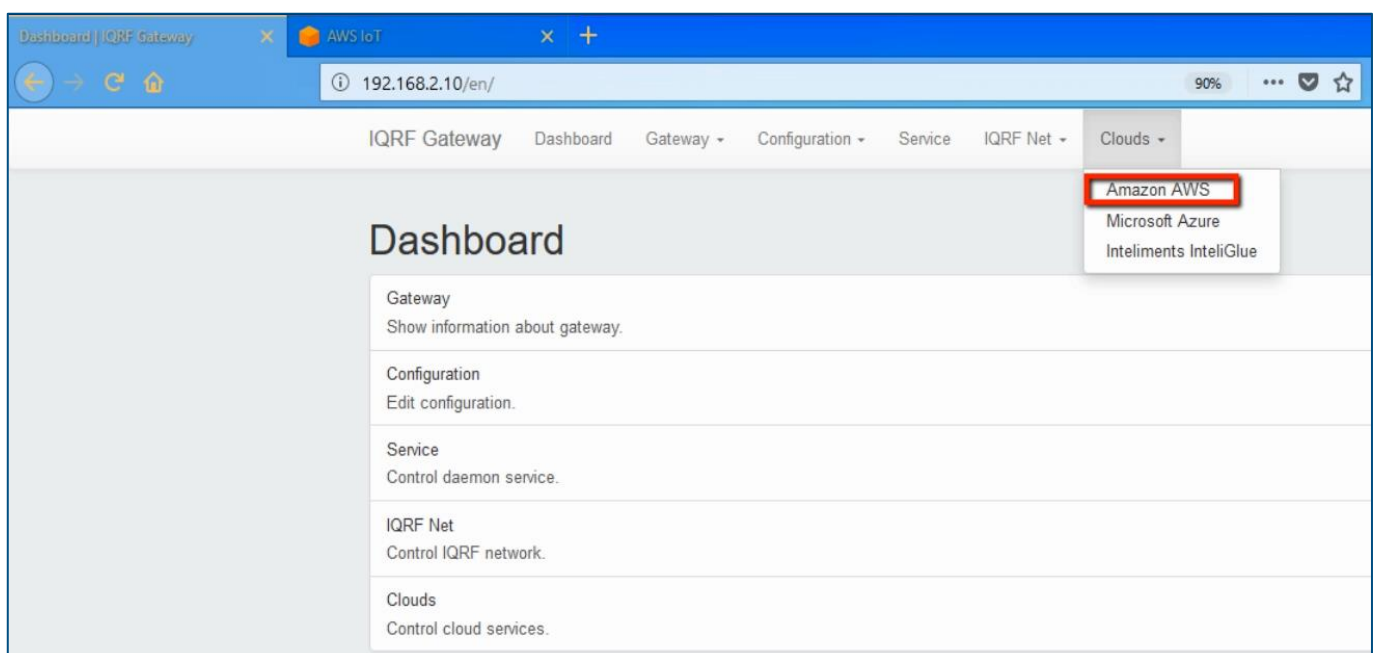


In the **Settings**, note the name of your **endpoint** because you will need it for the UP board configuration.



Files **rootCA.pem** (root certificate), **IQRF_Gateway.private.key** (private key file), and **IQRF_Gateway.cert.pem** (certificate file) should be already unzipped. We will transfer them to the UP board through the IQRF Gateway Daemon web application.

In the web browser on your computer, insert the IP address of your UP board, and login to it as *admin* with password *iqrf*. In the **IQRF Gateway Daemon web application**, click on the **Amazon AWS** item in the **Clouds** menu.



Enter the name of the **Endpoint** (find it in Settings of your AWS IoT). Select **rootCA.pem** as a Root CA certificate, **IQRF_Gateway.cert.pem** as a Certificate and **IQRF_Gateway.private.key** as a Private key file. Save the configuration.

Note: If you named your virtual device in AWS with a different name, names of files contain this name instead of IQRF_Gateway.

Add new MQTT interface

Endpoint
a24aso63wetb60.iot.eu-central-1.amazonaws.com

Root CA certificate
Select rootCA.pem file
ice_package\rootCA.pem Procházet...

Certificate
Select IQRF_Gateway.cert.pem - certificate file
IQRF_Gateway.cert.pem

Private key
Select IQRF_Gateway.private.key - private key file
IQRF_Gateway.private.key

Save

Inspect the new MQTT interface.

MQTT interface

Name	Broker	Client ID	TLS	Enabled	Edit	Remove
MqttMessaging1	tcp://127.0.0.1:1883	Local-app	✓	✓		
MqttMessaging2	ssl://a24aso63wetb60.iot.eu-central-1.amazonaws.com:8883	IQRF-GW-test	✓	✓		

Add

Edit the second MQTT interface

Address of the **endpoint** goes after the **SSL** protocol and at the end of Broker address is the port number **8883**. **Iqrf/DpaRequest** is set as the topic for commands, and **Iqrf/DpaResponse** is set as the topic for responses.

The image shows a screenshot of the 'Edit MQTT interface' configuration page. The page contains several fields and checkboxes, with callouts pointing to specific values:

- Name:** MqttMessaging2 (Callout: Name of the MQTT interface)
- Enabled:**
- Broker address:** ssl://a24aso63wetb60.iot.eu-central-1.amazonaws.com:8883 (Callout: Endpoint name and port)
- Client ID:** IQRF-GW-test (Callout: Client ID)
- Persistence:** 1
- QoS:** QoS 1 - At least once
- Topic for requests:** Iqrf/DpaRequest (Callout: Commands)
- Topic for responses:** Iqrf/DpaResponse (Callout: Responses)
- User:** (empty field)
- Password:** (empty field)
- Enable TLS:**
- Keep alive interval:** 20

There are the **timeout**, the **minimum**, and **maximum** connections set, and the path to the uploaded files that set up a secure connection between the gateway and the cloud. Check the **Enable server certificate authentication** item.

Connect timeout
5

Min reconnect
1

Max reconnect
64

Trust store
/etc/iqrf-daemon/certs/2018-02-08T17:18:08+0100aws-ca.crt
path to uploaded rootCA.pem file

Key store
/etc/iqrf-daemon/certs/2018-02-08T17:18:08+0100-aws.crt
path to uploaded IQRf_Gateway.cert.pem file

Private key
/etc/iqrf-daemon/certs/2018-02-08T17:18:08+0100-aws.key
path to uploaded IQRf_Gateway.private.key file

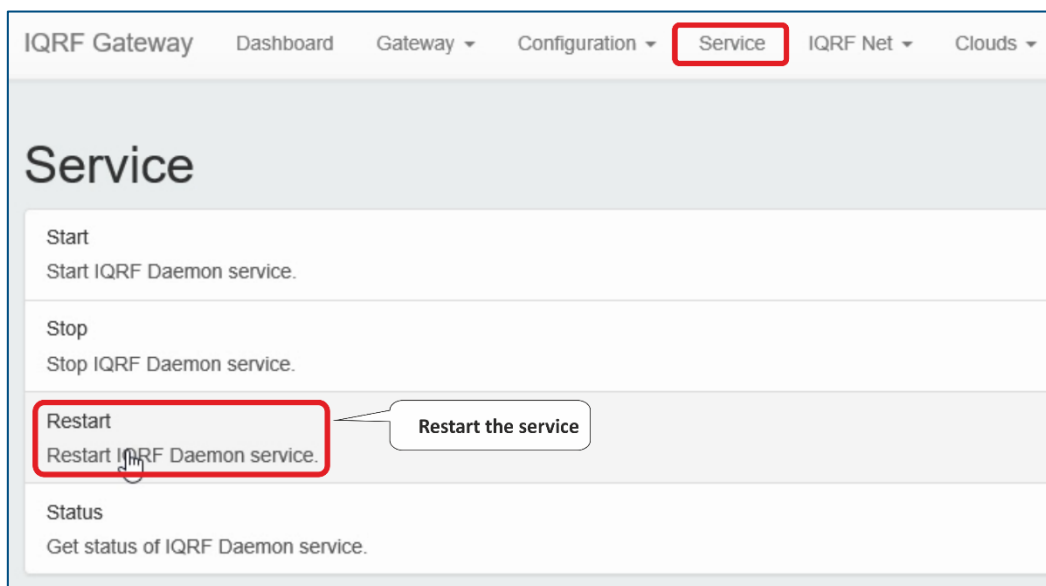
Private key password
[Empty field]

Enabled cipher suites
[Empty field]

Enable server certificate authentication

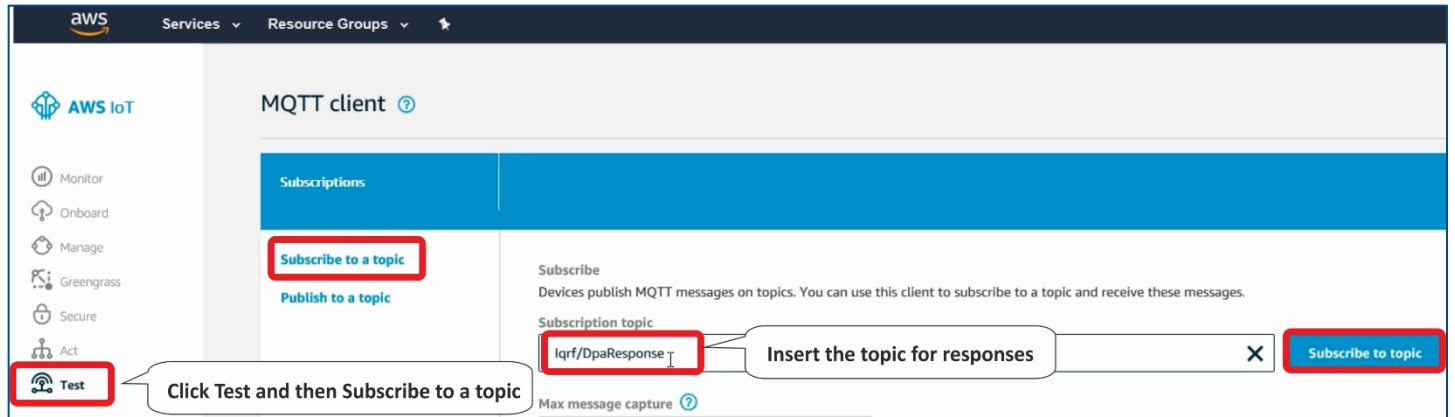
Save

Restart IQRf Gateway Daemon. After restarting, check the status of the UP board if the selected services are running.



4 Test the connection

In the web browser on your computer, in AWS IoT, click **Test**. Enter the **Iqrf/DpaResponse** to the Response topic to retrieve the gateway responses and click on **Subscribe to topic**.



To send commands from the cloud to the gateway, set the **Iqrf/DpaRequest** as the topic for requests. Gateway will expect commands in this topic.



Insert a DPA packet in the JSON format into the text box and click on **Publish to topic**. In our example, we sent a command to turn on the red LED on the coordinator.

```
{
  "ctype": "dpa",
  "type": "raw",
  "msgid": "1510754980",
  "request": "00.00.06.01.FF.FF",
  "request_ts": "",
  "confirmation": "",
  "confirmation_ts": "",
  "response": "",
  "response_ts": ""
}
```


We can see that the gateway picked up and executed the command, and sent a confirmation with "No Error" into the **Iqrf/DpaResponse** topic.

Publish

Specify a topic and a message to publish with a QoS of 0.

Iqrf/DpaRequest Publish to topic

```

3  "type": "raw",
4  "msgid": "1510754980",
5  "request": "00.00.06.01.FF.FF",
6  "request_ts": "",
7  "confirmation": "",
8  "confirmation_ts": "",
9  "response": "",
10 "response_ts": ""
11 }
    
```

DPA command in JSON format

Iqrf/DpaResponse Export HI

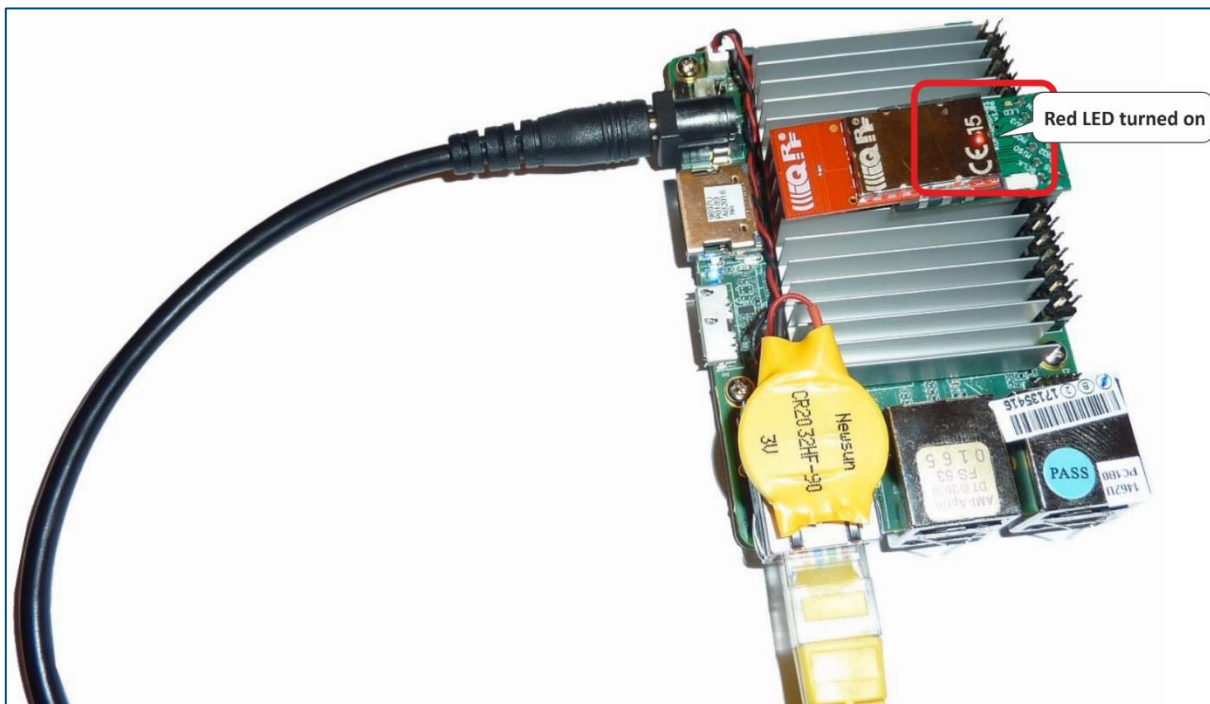
Nov 21, 2017 8:30:52 AM +0100

```

{
  "ctype": "dpa",
  "type": "raw",
  "msgid": "1510754980",
  "request": "00.00.06.01.ff.ff",
  "request_ts": "2017-11-21 07:30:52.71517",
  "confirmation": "",
  "confirmation_ts": "",
  "response": "00.00.06.81.00.00.00.00",
  "response_ts": "2017-11-21 07:30:52.119054",
  "status": "STATUS_NO_ERROR"
}
    
```

Response

We can visually double check the result of this command. The red LED turned on.



5 Summary

The bidirectional communication between IQRF network and the Amazon Web Services is up and running. Now it's just up to you to use it for your own IoT solution.