

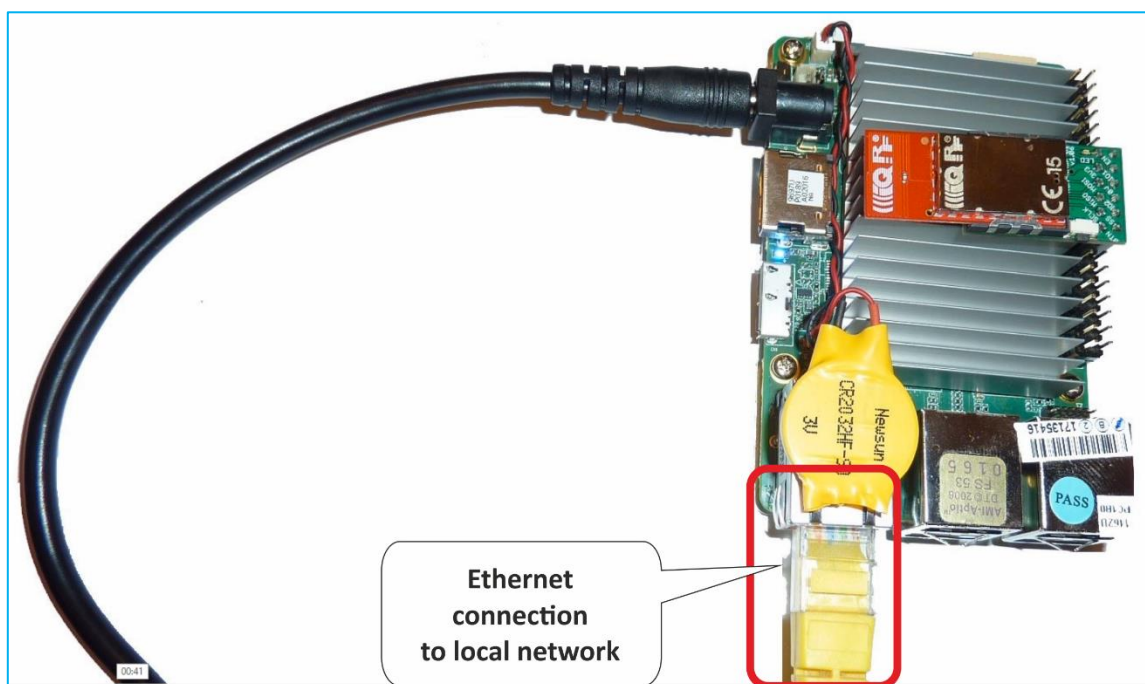
# IoT Starter Kit – Part 3a:

## How to connect to Amazon Web Services

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.

### 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 and configure your gateway through the IQRF Daemon web application.



## AWS account

First, create an Amazon Web Services account ([aws.amazon.com](https://aws.amazon.com)). You have to 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.



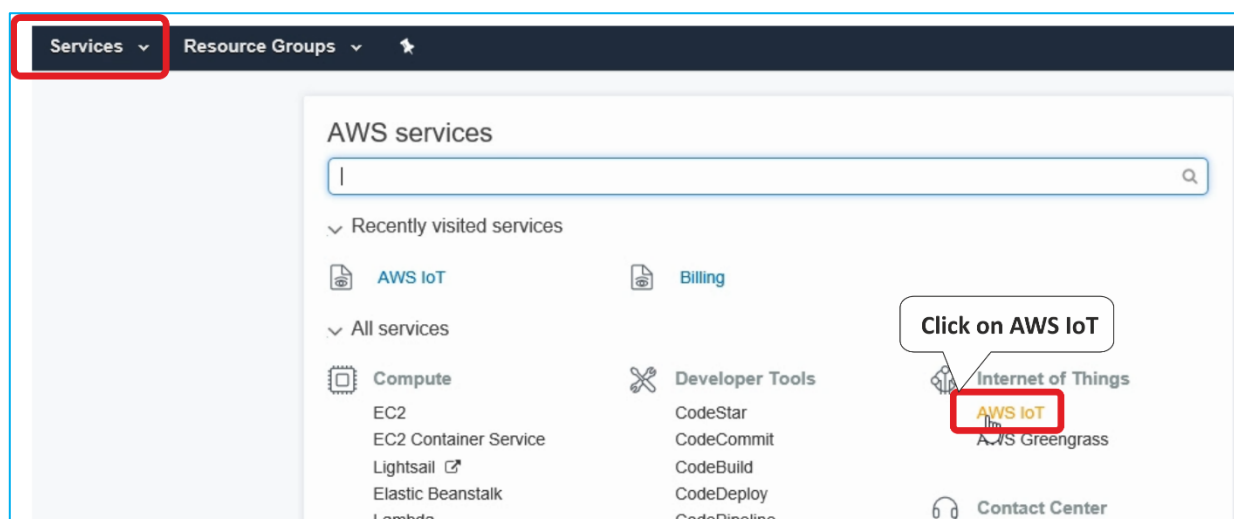
The screenshot shows the AWS homepage with the 'Create an AWS Account' button highlighted in the top right navigation bar. A green arrow points down to the 'Create a new AWS Account' sign-up page. The sign-up page includes a form with the following fields:

- AWS account name:** IGRP-test-connection
- Email address:** alliance@igrt.org
- Password:** [masked]
- Confirm password:** [masked]

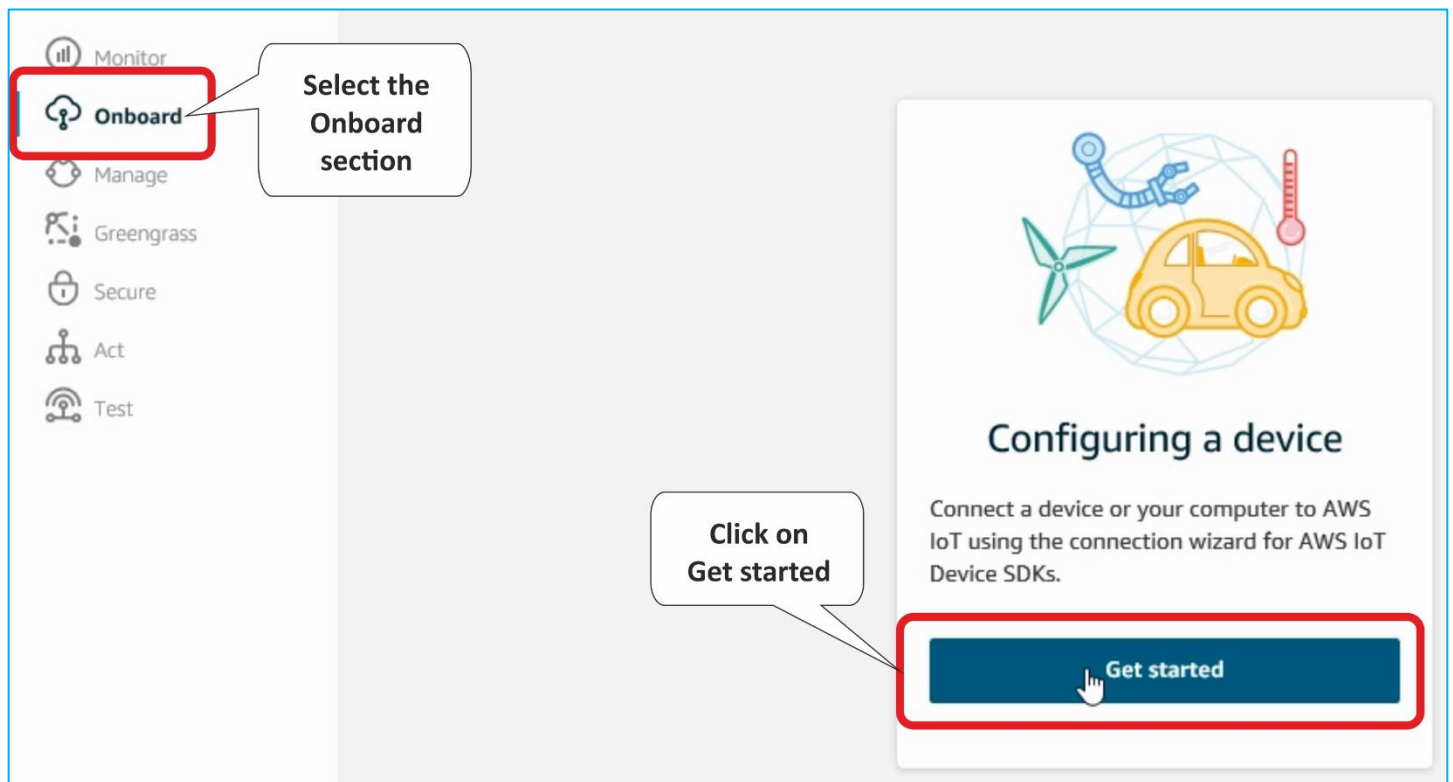
The 'Continue' button is highlighted in yellow. To the right of the form, there is a graphic of a laptop with a checkmark and the text: 'AWS Accounts Include 12 Months of Free Tier Access'. Below this, it lists 'Including use of Amazon EC2, Amazon S3, and Amazon DynamoDB' and provides a link to 'Visit aws.amazon.com/free for full offer terms'.

## Setup the connection to AWS

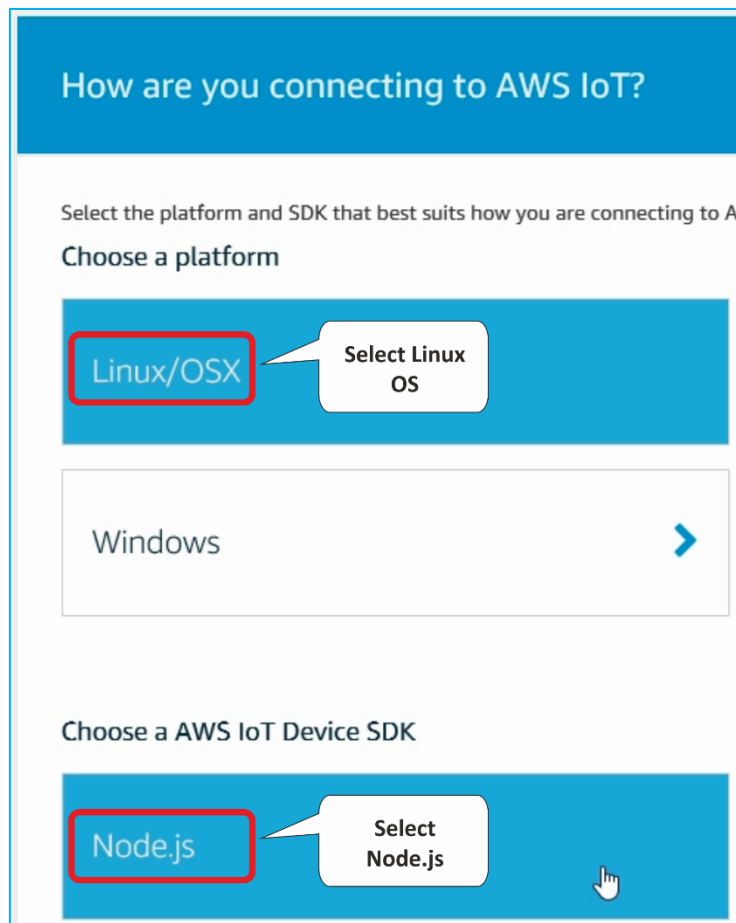
In **Services**, in the **Internet of Things** section, find **AWS IoT**.



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.

CONNECT TO AWS IOT

Register a thing

STEP 1/3

A thing is the representation and record of your physical device in the cloud. Any physical device needs a thing to work with AWS IoT. Creating a thing will also create a thing shadow.

[Choose an existing thing instead?](#)

Name

IQRF\_Gateway

Name your device

Show optional configuration (this can be done later) ▾

Back

Next step

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

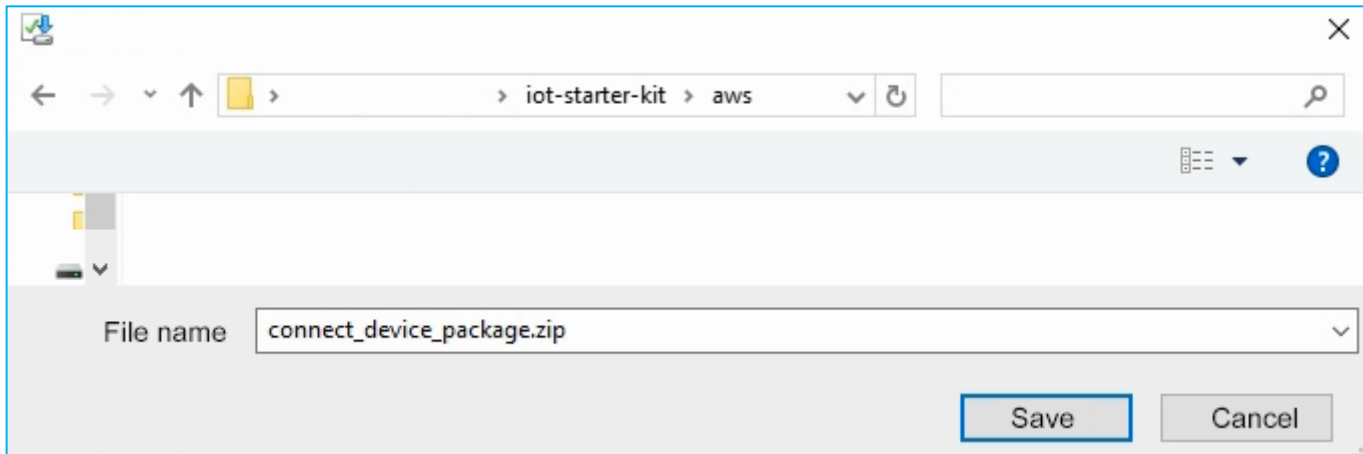
Before your device can connect and publish messages, you will need to download the connection kit.

Download connection kit for

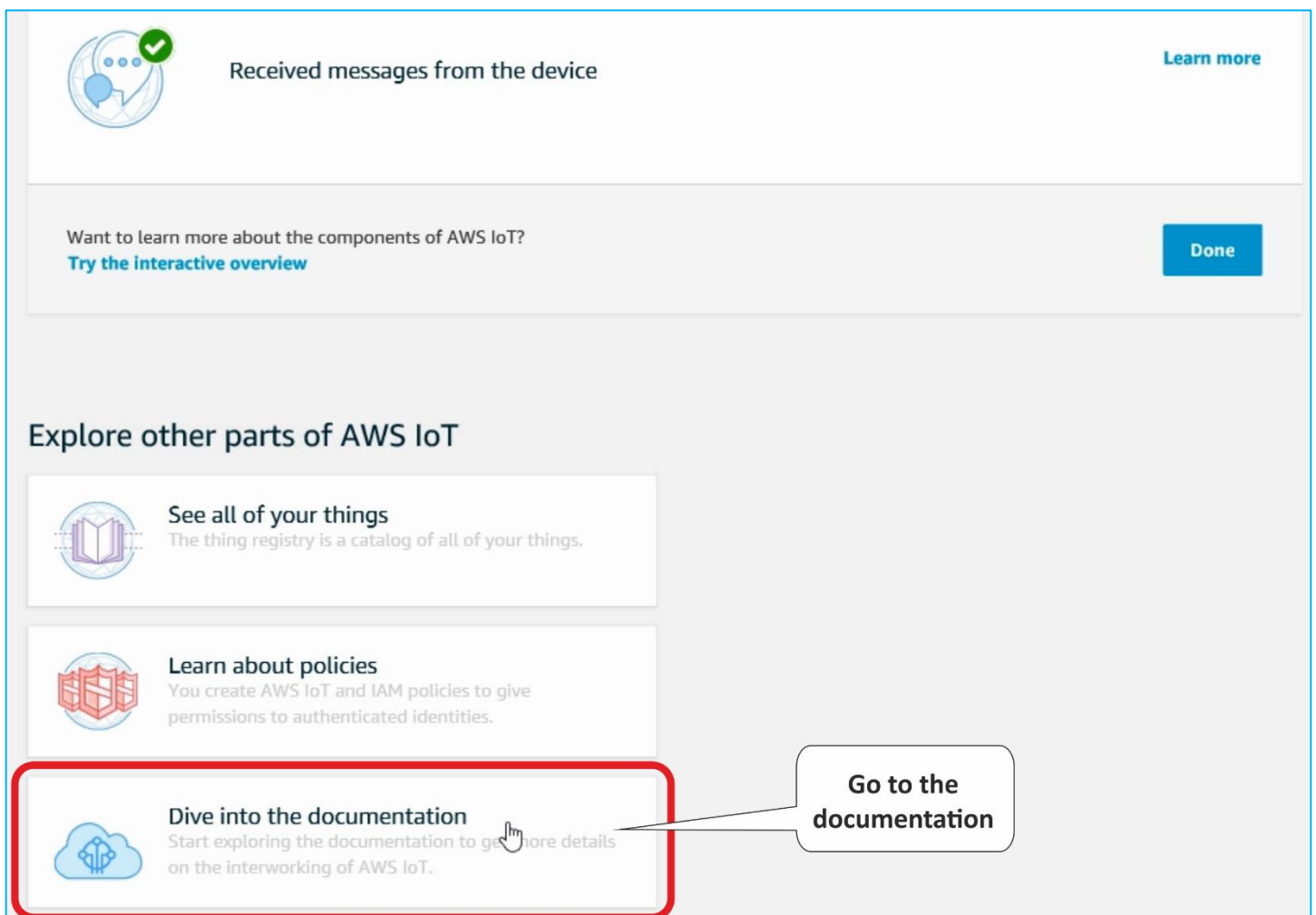
Linux/OSX

Download connection kit

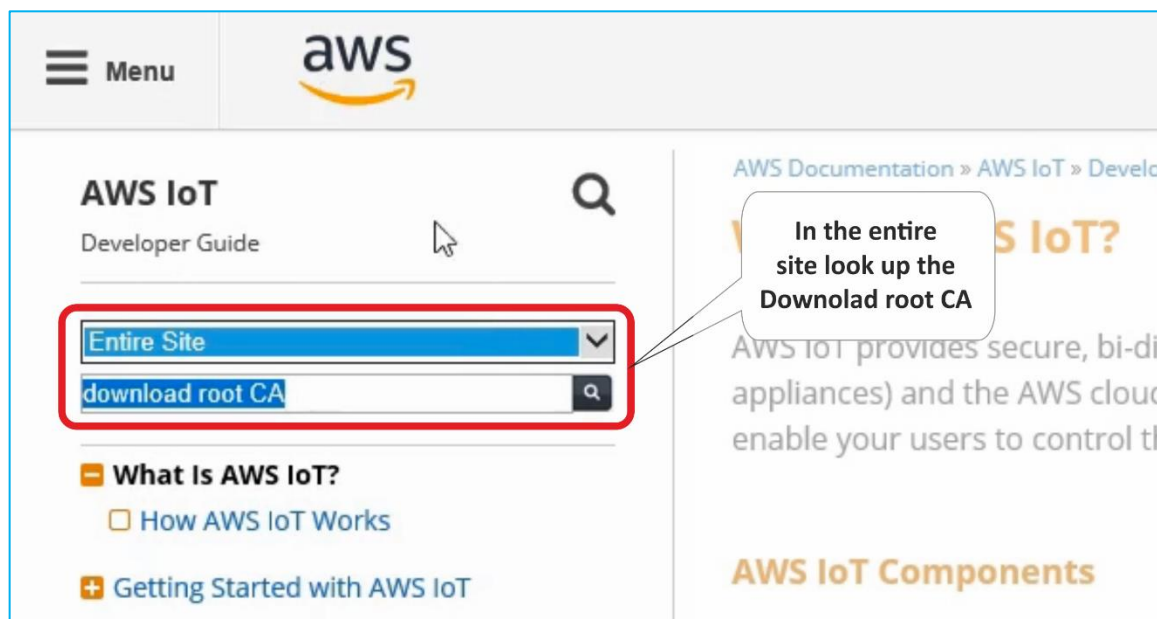
Save and unzip this file. Store the certificate and the keys for further use.



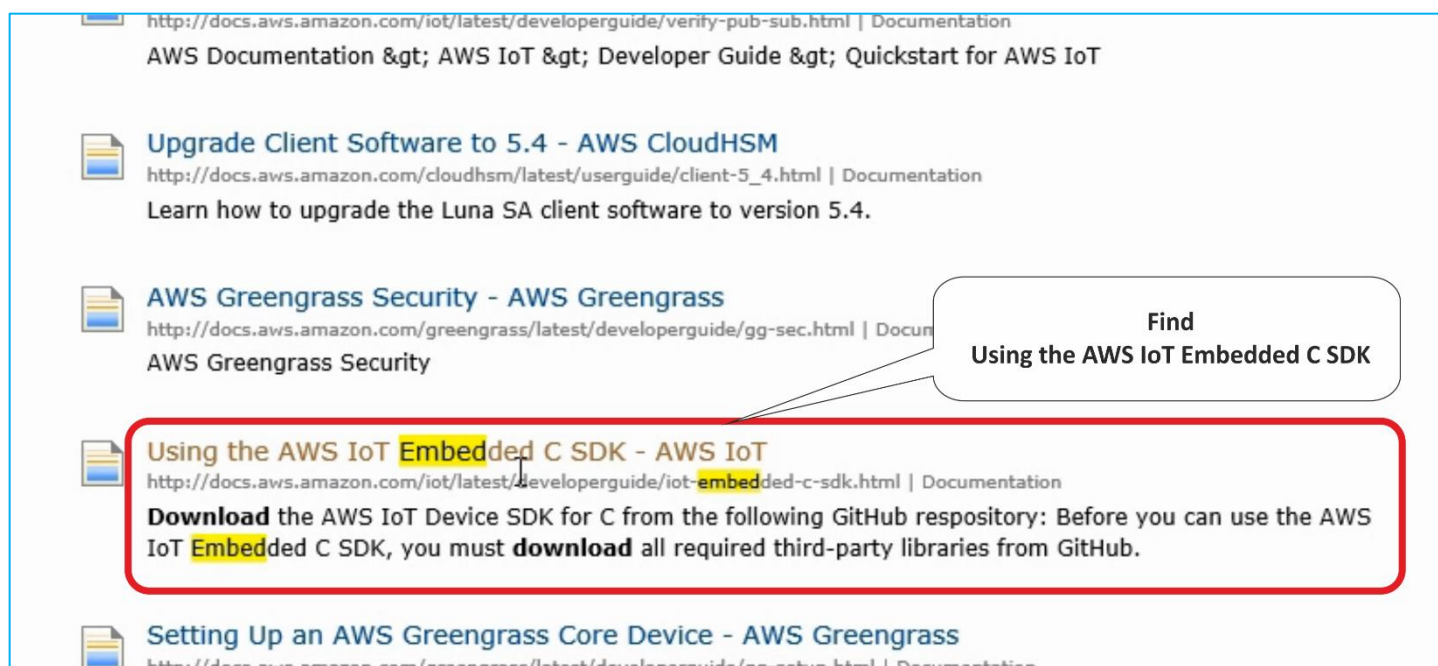
After saving process go to the documentation.



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.





Here you can find the root certificate.

## Using the AWS IoT Embedded C SDK

## Set Up the Runtime Environment for the AWS IoT Embedded C SDK

1. Download the AWS IoT Device SDK for C from the following GitHub repository:

```
git clone https://github.com/aws/aws-iot-device-sdk-embedded-C.git -b release
```

2. Before you can use the AWS IoT Embedded C SDK, you must download all required third-party libraries from GitHub. You can find these libraries in the `deviceSDK/external_libs` folder.

## Sample App Configuration

The AWS IoT Embedded C SDK includes sample apps for you to try. For simplicity, we are going to run `subscribe_publish_sample`.

1. Copy your certificate, private key and root CA certificate into the `deviceSDK/certs` directory.

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 the `deviceSDK/certs` directory.

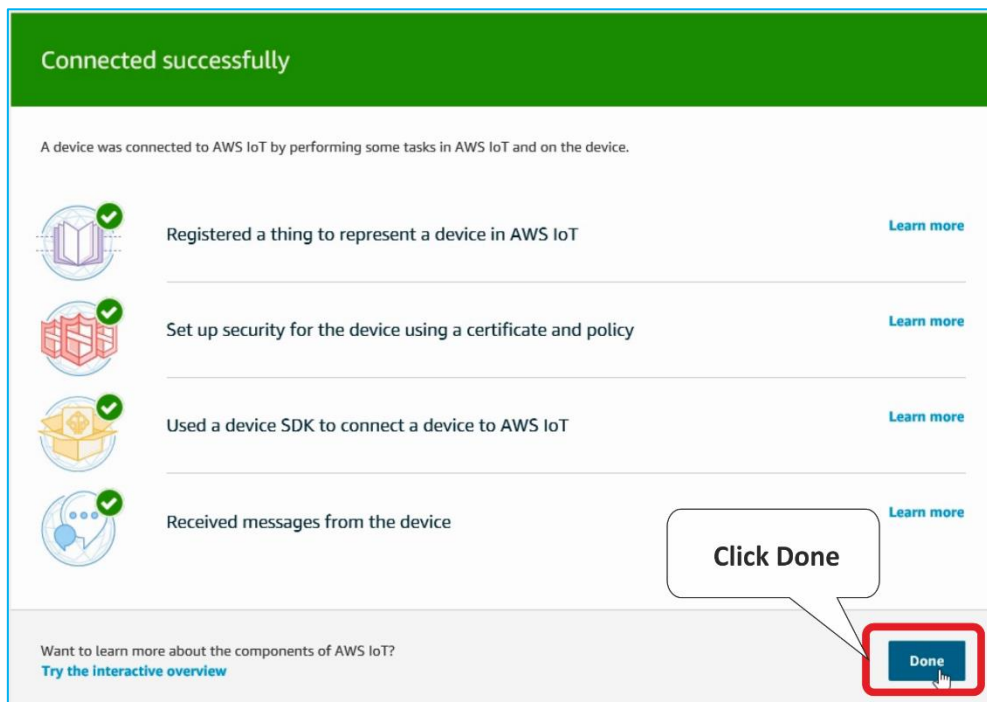
## Root CA certificate

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

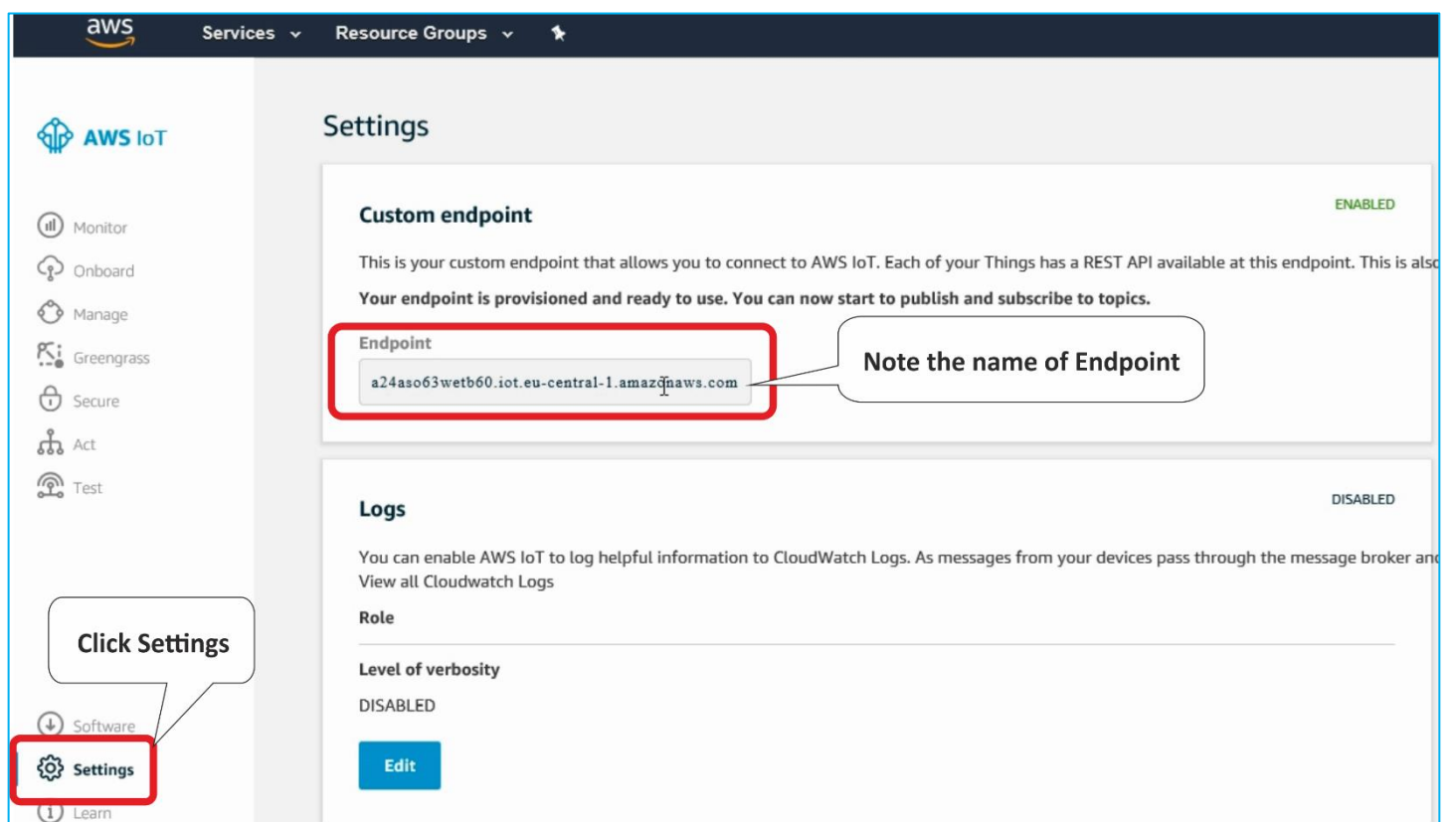
Note: You can choose your own name but in later steps, you need to use the given name.



Next, click Done.



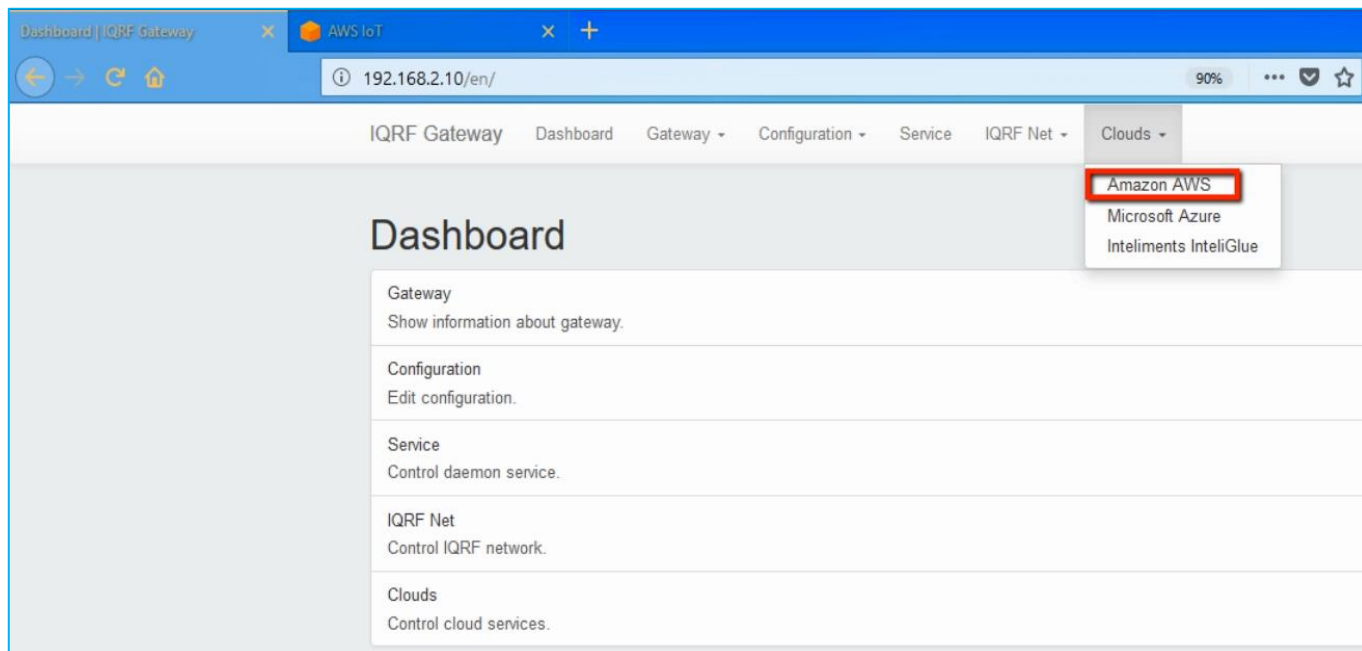
In the **Settings**, note the name of your **endpoint** because you will need it for the UP board configuration.





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 enter the IP address of your UP board. In the **IQRF Gateway Daemon web application**, in the **Clouds** menu click on the **Amazon AWS** item.



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 another 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**

ice\_package\rootCA.pem Procházet... Select rootCA.pem file

**Certificate**

IQRF\_Gateway.cert.pem Select IQRF\_Gateway.cert.pem - certificate file

**Private key**

IQRF\_Gateway.private.key Select IQRF\_Gateway.private.key - private key file

**Save**

Inspect the new MQTT interface.

### MQTT interface

Edit the second MQTT interface

Name	Broker	Client ID	TLS	Enabled	Edit	Remove
MqttMessaging1	tcp://127.0.0.1:1883	Local-app		<input checked="" type="checkbox"/>		
MqttMessaging2	ssl://a24aso63wetb60.iot.eu-central-1.amazonaws.com:8883	IQRF-GW-test	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

Add

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.

### Edit MQTT interface

Name

MqttMessaging2

Name of the MQTT interface

☒ Enabled

Broker address

ssl://a24aso63wetb60.iot.eu-central-1.amazonaws.com:8883

Endpoint name and port

Client ID

IQRF-GW-test

Client ID

Persistence

1

QoS

QoS 1 - At least once

Topic for requests

Iqrf/DpaRequest

Commands

Topic for responses

Iqrf/DpaResponse

Responses

User

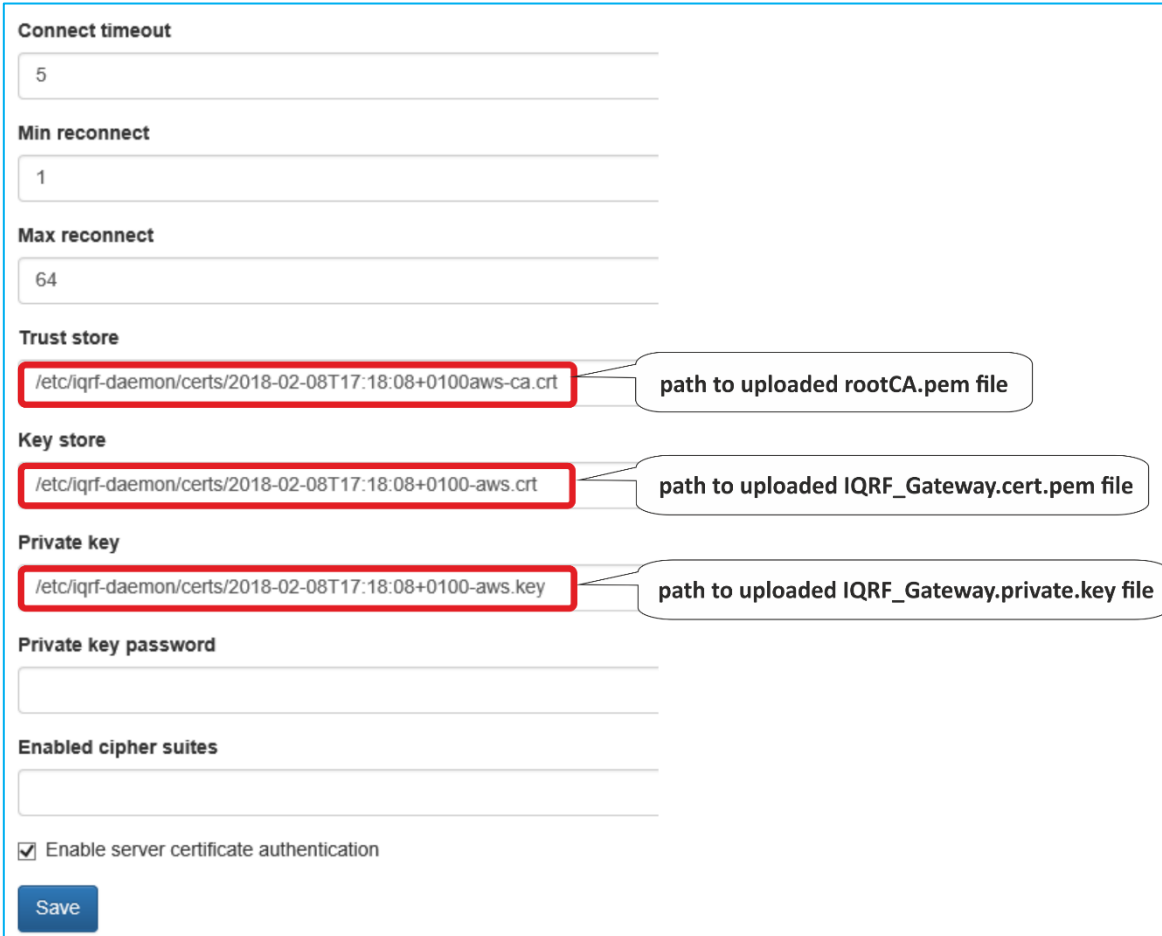
Password

☒ 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

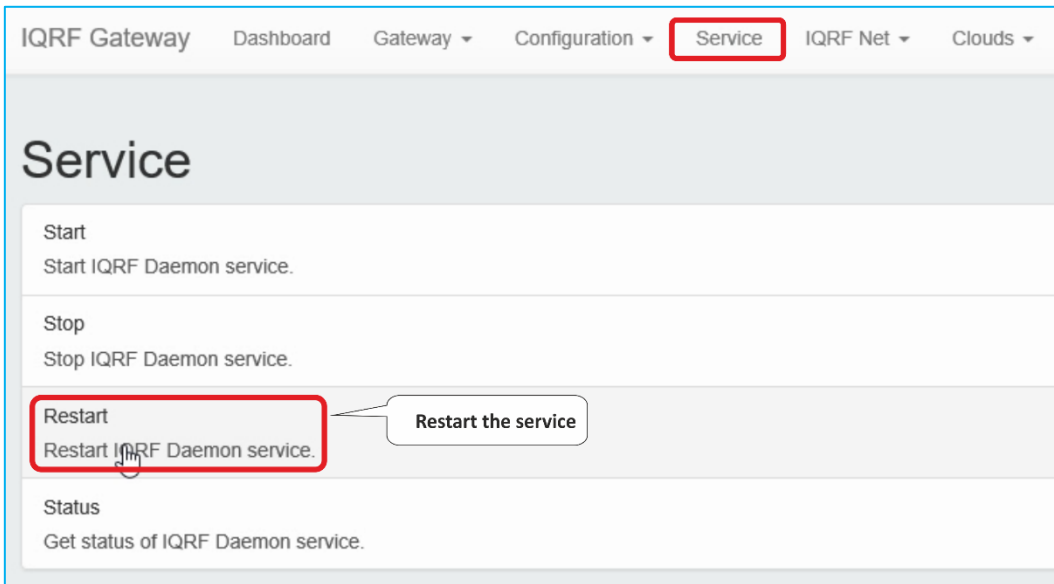
**Private key password**

**Enabled cipher suites**

☒ Enable server certificate authentication

**Save**

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



**IQRF Gateway** Dashboard Gateway Configuration **Service** IQRF Net Clouds

## Service

**Start**  
Start IQRF Daemon service.

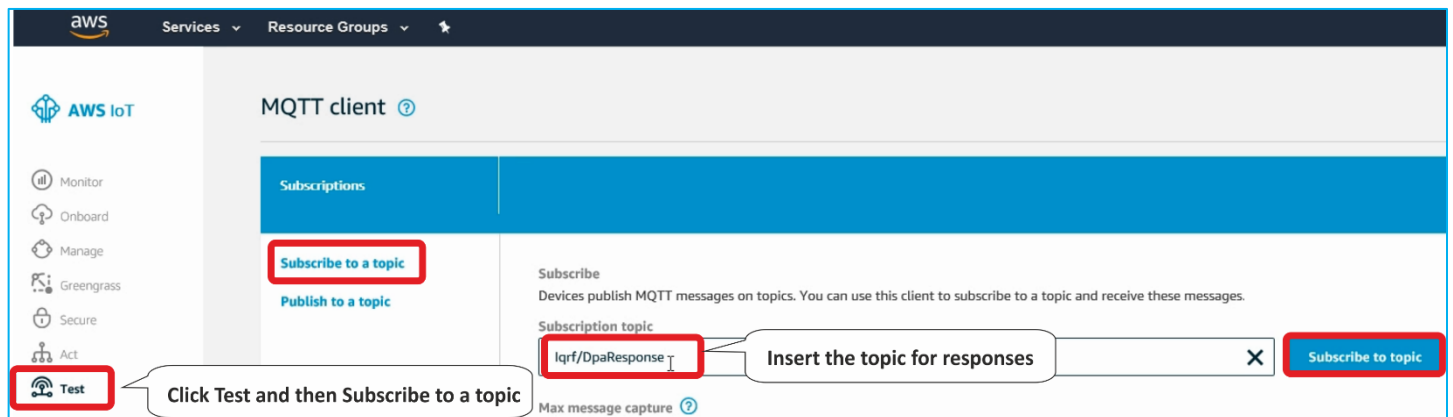
**Stop**  
Stop IQRF Daemon service.

**Restart**  
Restart IQRF Daemon service.  
Restart the service

**Status**  
Get status of IQRF Daemon service.

## Test the connection

In the web browser on your computer, in AWS IoT, click **Test**. Enter the **lqrf/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 **lqrf/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 **lqrf/DpaResponse** topic.

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

lqrf/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

**lqrf/DpaResponse** Nov 21, 2017 8:30:52 AM +0100 Export HI

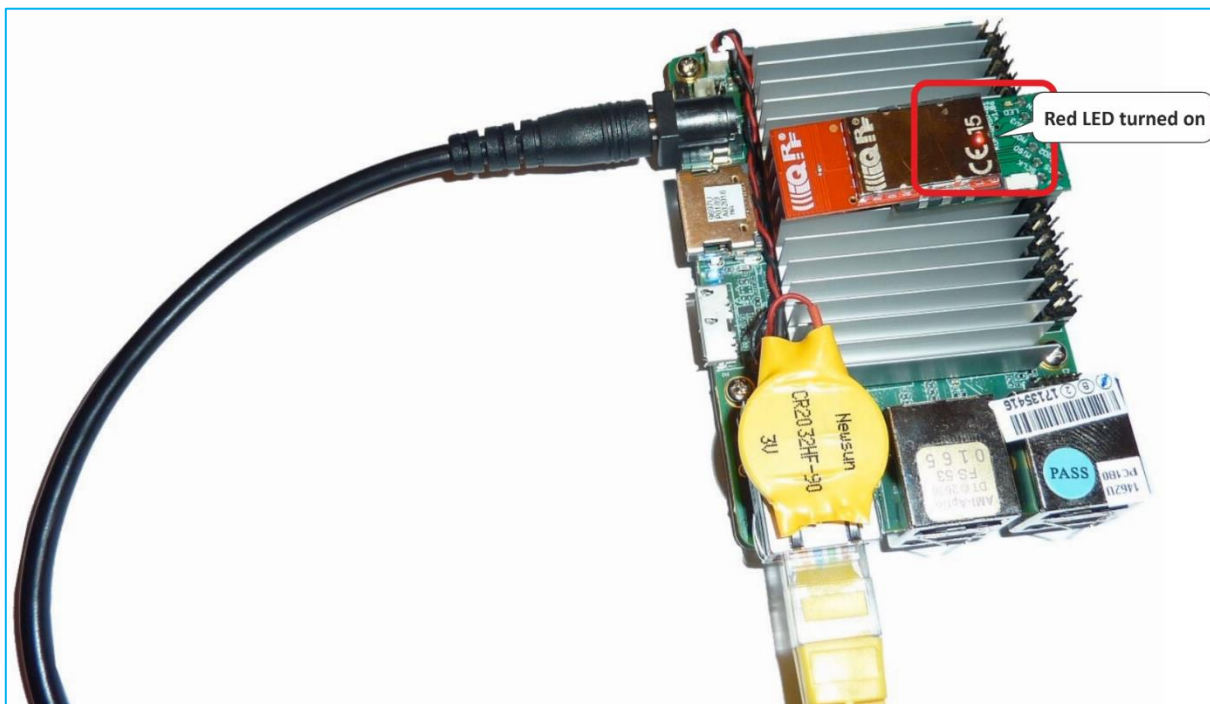
```

{
  "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.



## 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.