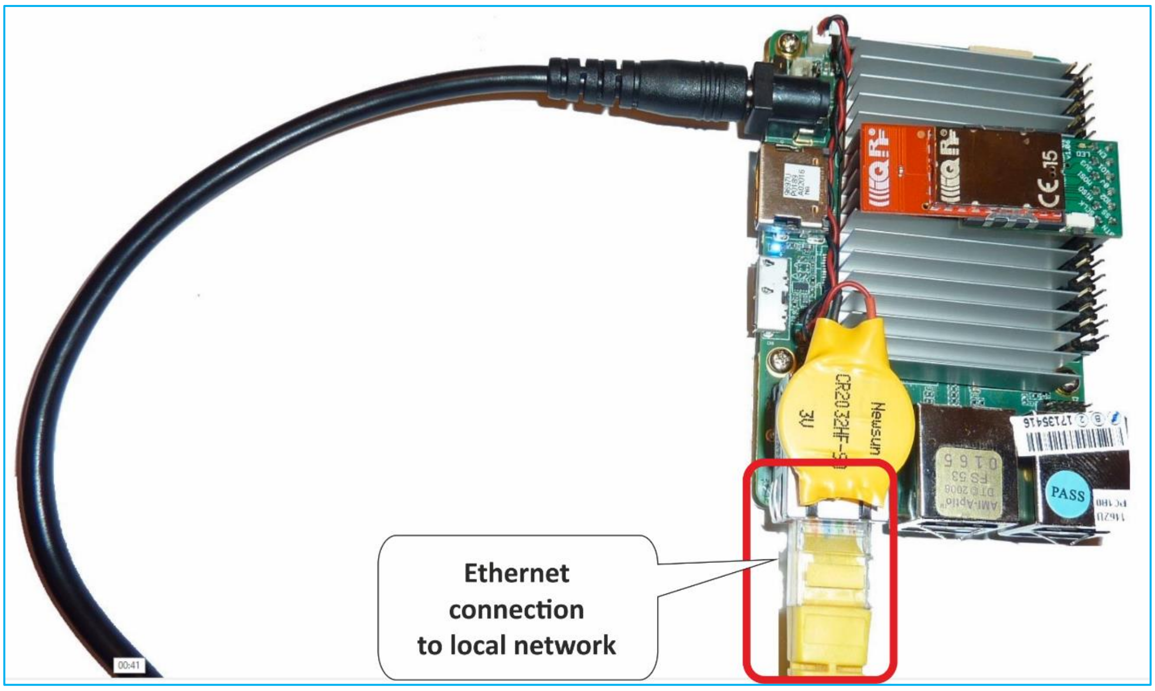
IoT Starter Kit – Part 3a: How to connect to Amazon Web Services

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.

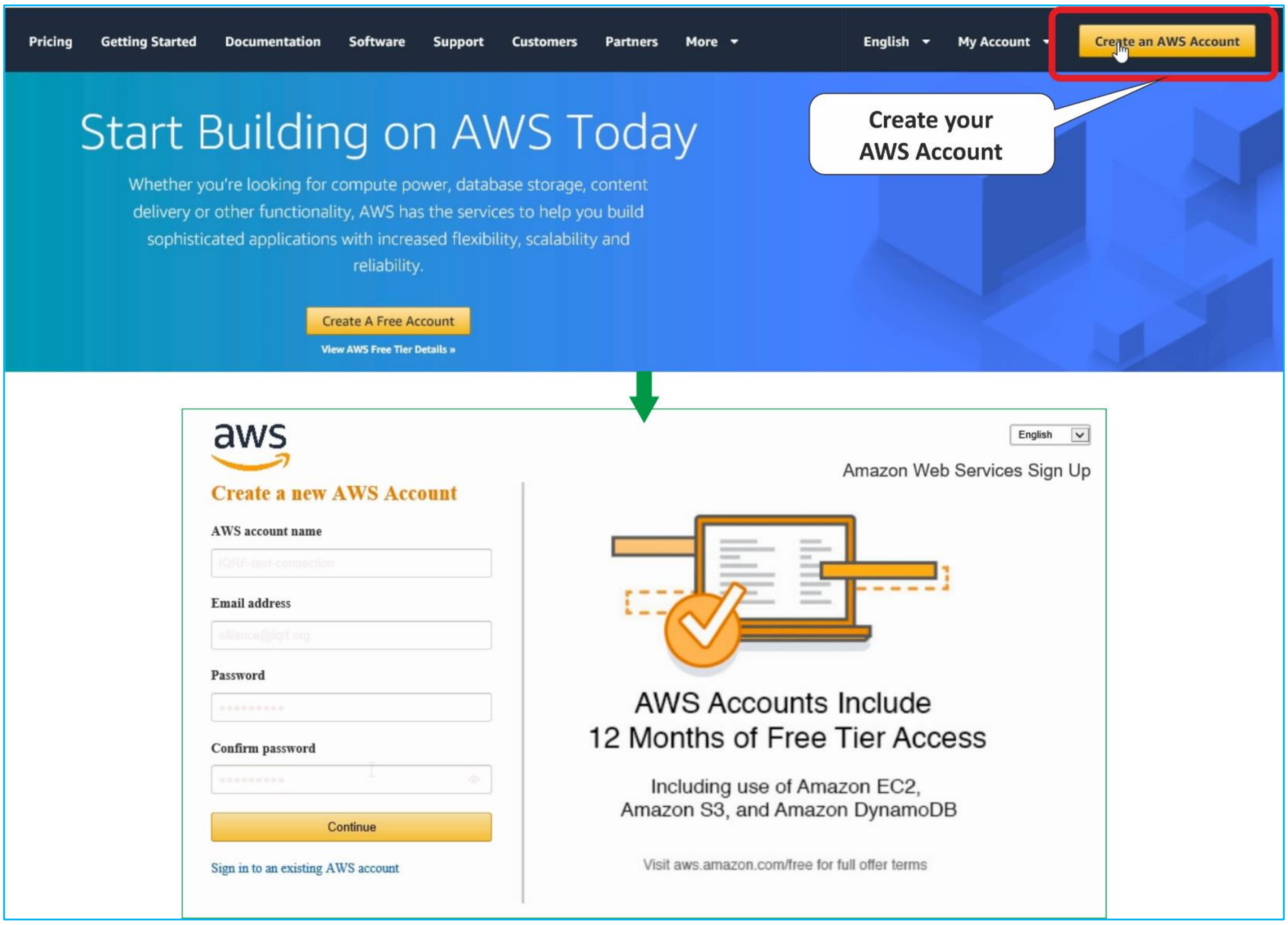
Local network

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 and configure your gateway through the IQRF Daemon web application.



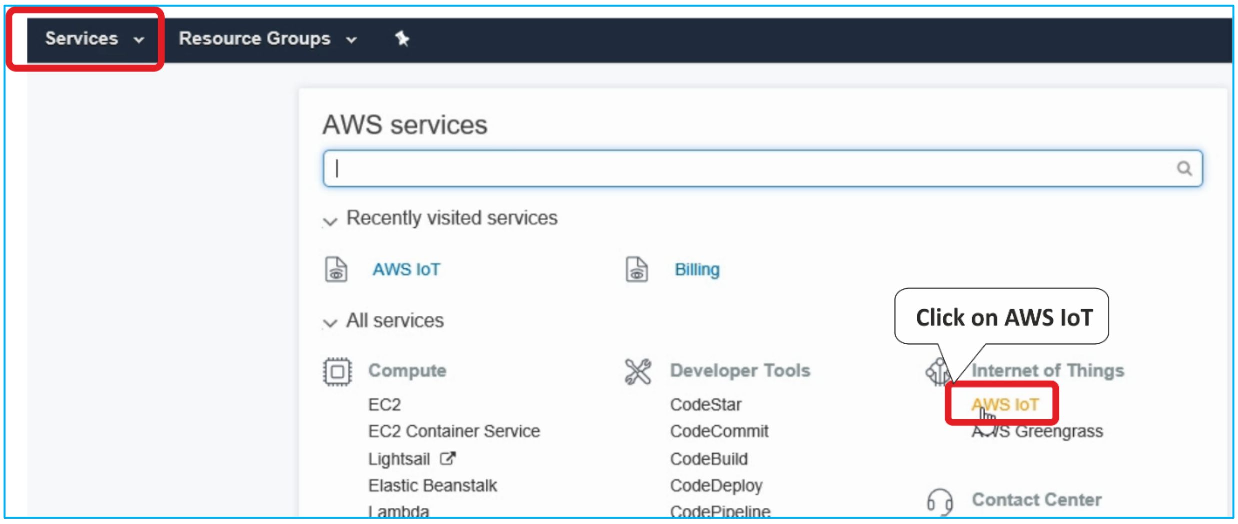
AWS account

First, create an Amazon Web Services account (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.

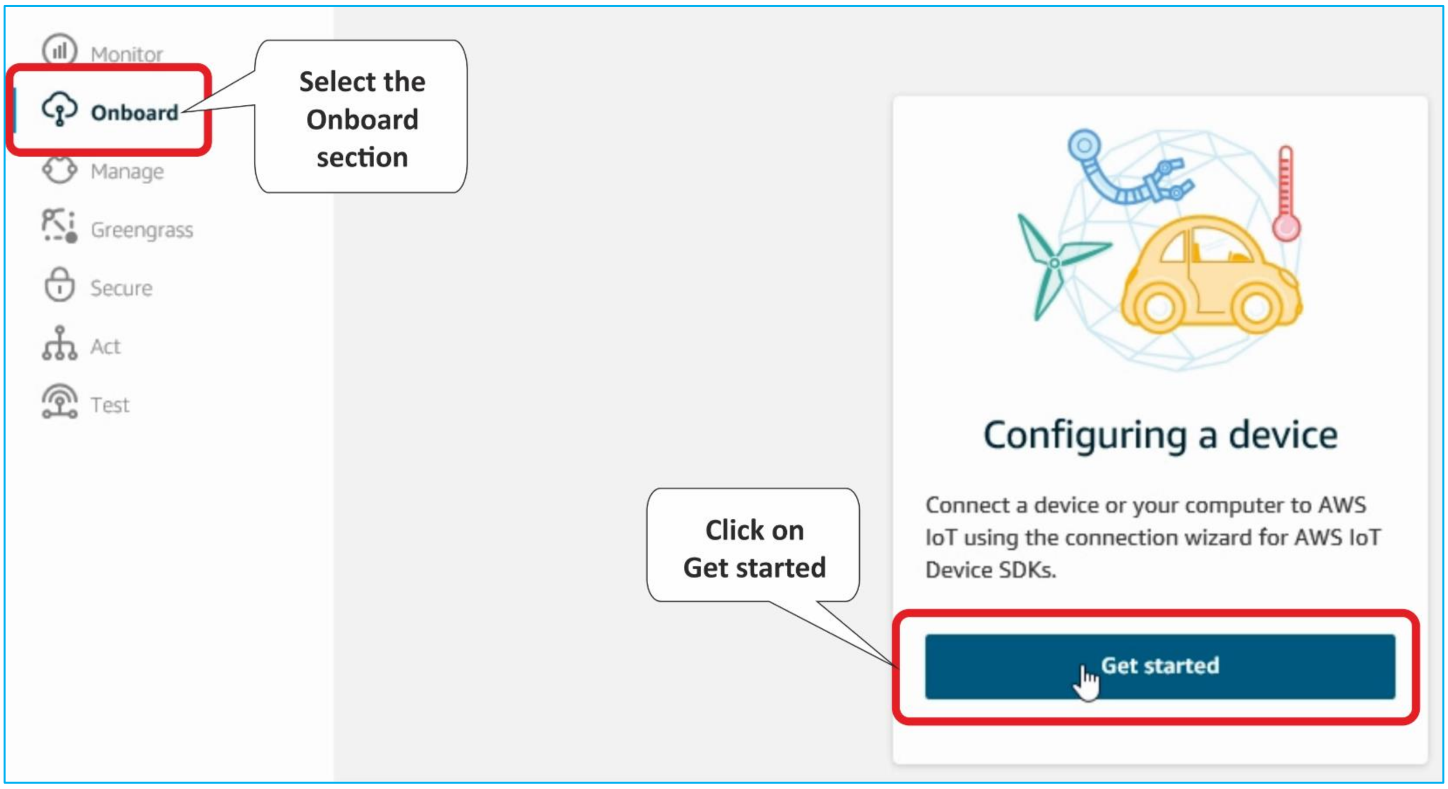

Pricing
Getting Started
Documentation
Software
Support
Customers
Partners
More
English
My Account
Create an AWS Account
Create your AWS Account
Start Building on AWS Today
Whether you're looking for compute power, database storage, content delivery or other functionality, AWS has the services to help you build sophisticated applications with increased flexibility, scalability and reliability.
Create A Free Account
View AWS Free Tier Details »
aws
English
Amazon Web Services Sign Up
Create a new AWS Account
AWS account name
Email address
Password
Confirm password
Continue
Sign in to an existing AWS account
AWS Accounts Include 12 Months of Free Tier Access
Including use of Amazon EC2, Amazon S3, and Amazon DynamoDB
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**.



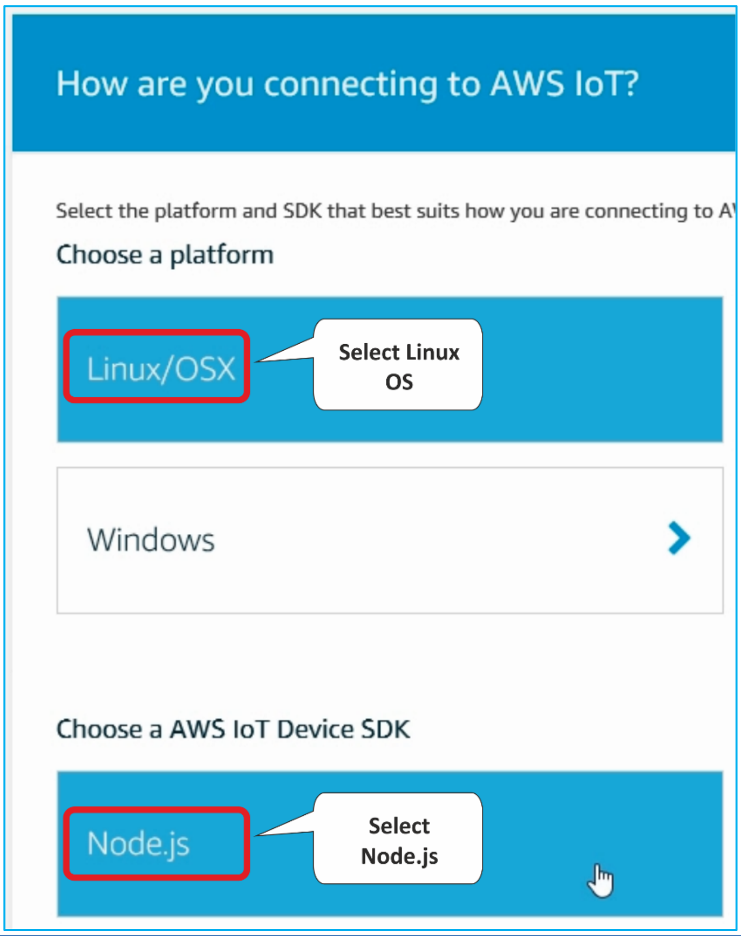
Click on **Get started** in the **Onboard** section. You will register your device, download the connection kit, and configure and test the connection with your device.



Configuring a device

Connect a device or your computer to AWS IoT using the connection wizard for AWS IoT Device SDKs.

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.



How are you connecting to AWS IoT?

Select the platform and SDK that best suits how you are connecting to A

Choose a platform

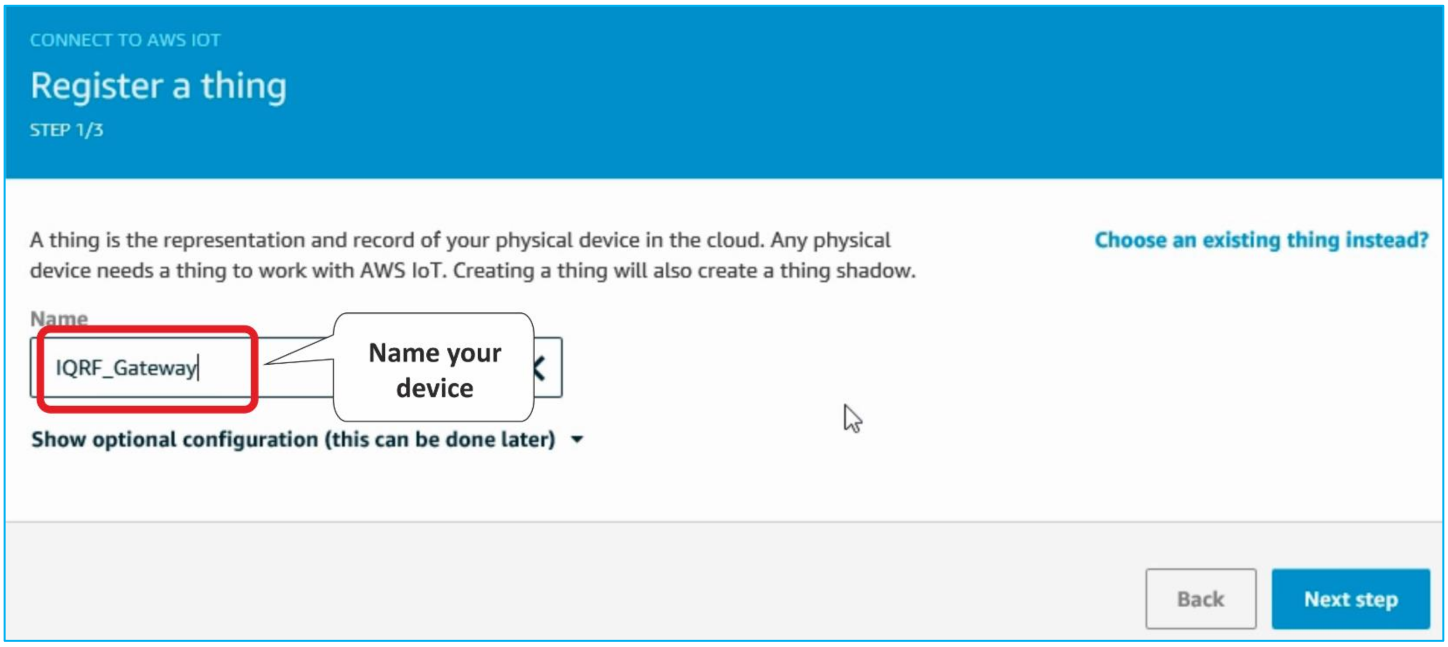
Linux/OSX

Windows

Choose a AWS IoT Device SDK

Node.js

Enter the name of your connected device.


CONNECT TO AWS IOT

Register a thing

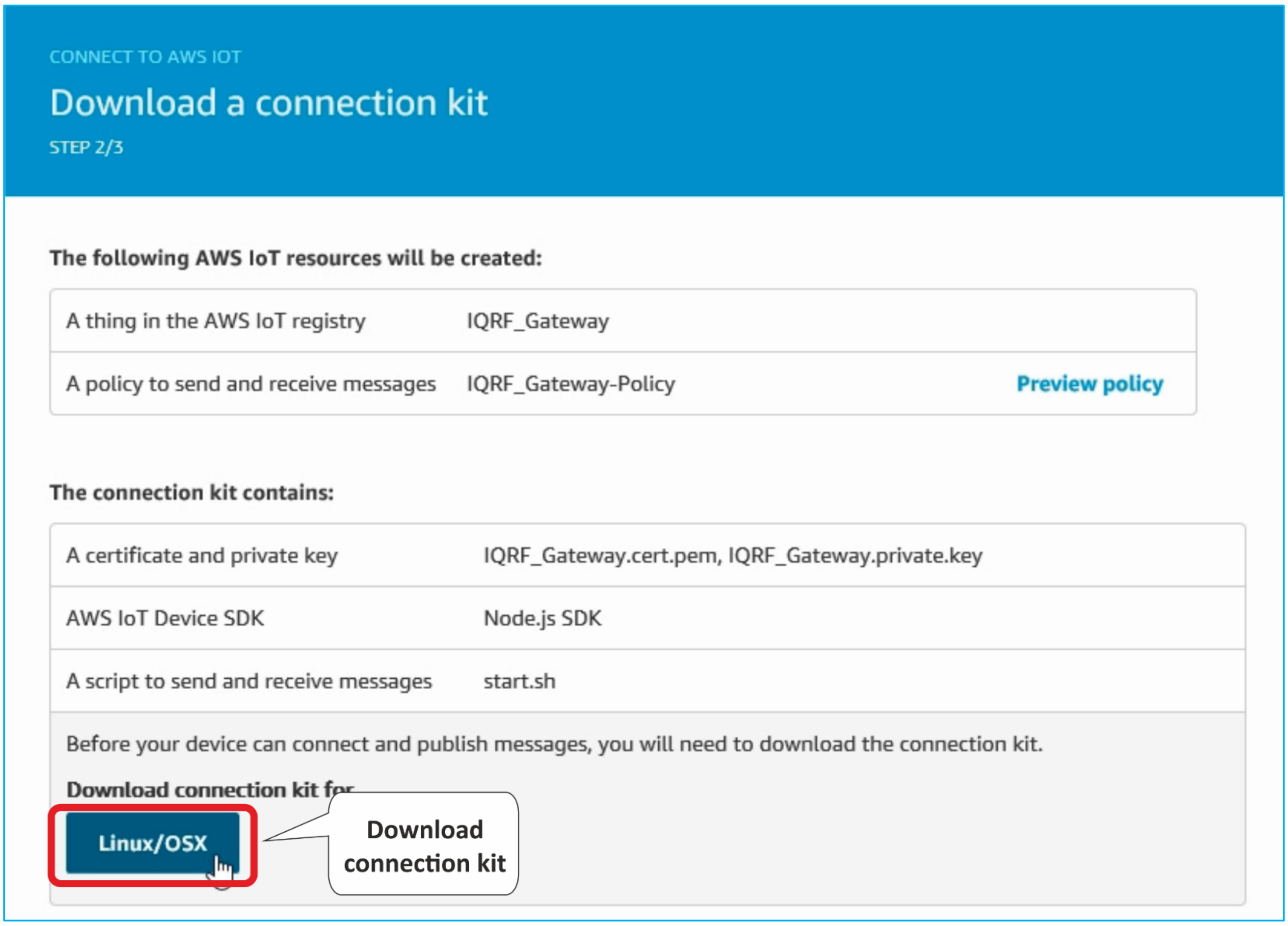
STEP 1/3

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.

Name: IQRF_Gateway

Choose an existing thing instead?
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

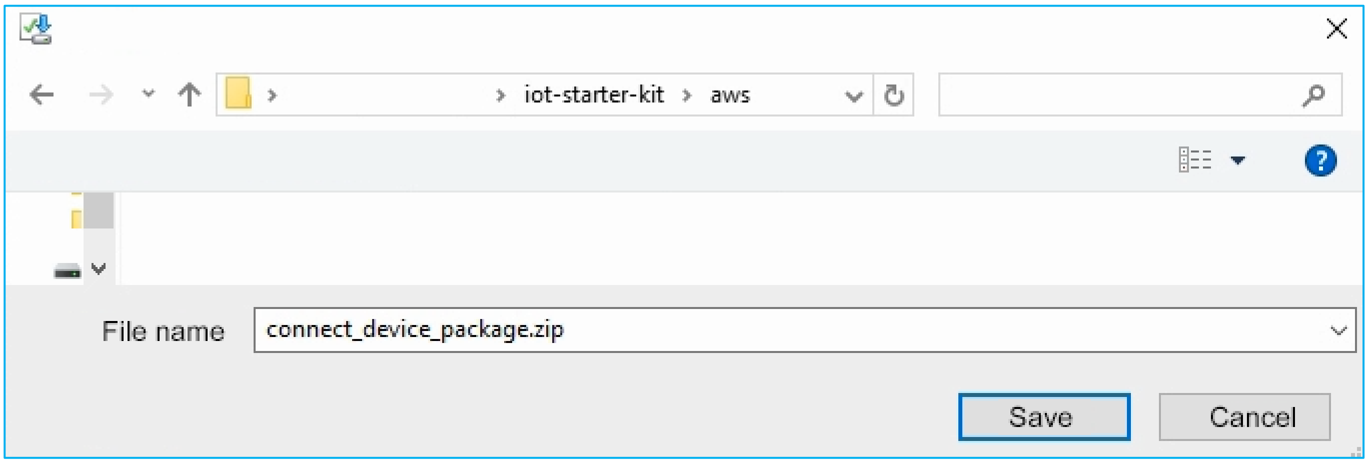
The connection kit contains:

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

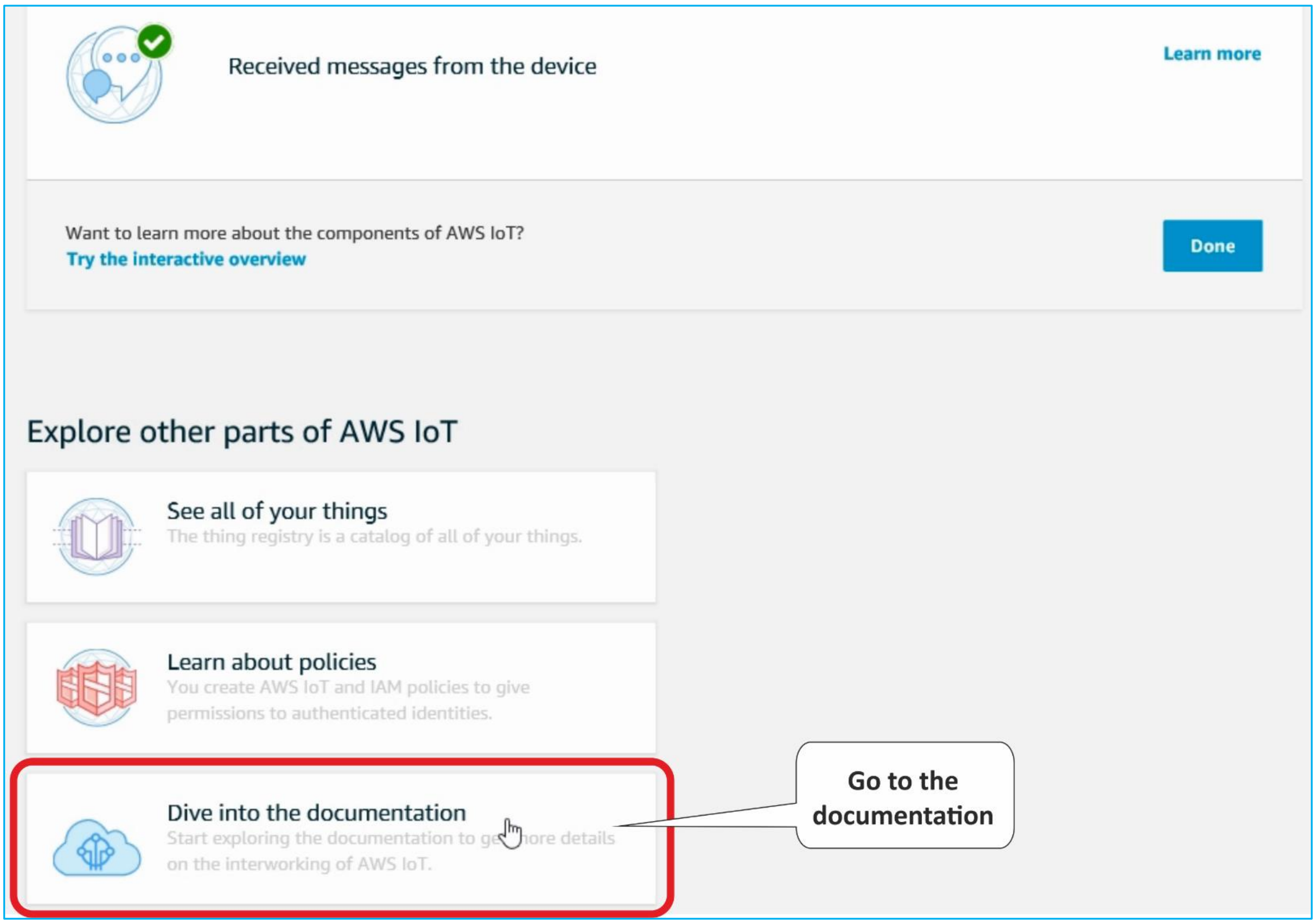
Before your device can connect and publish messages, you will need to download the connection kit.

Download connection kit for:

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



After saving process go to the documentation.


Received messages from the device

Learn more

Want to learn more about the components of AWS IoT?
Try the interactive overview

Done

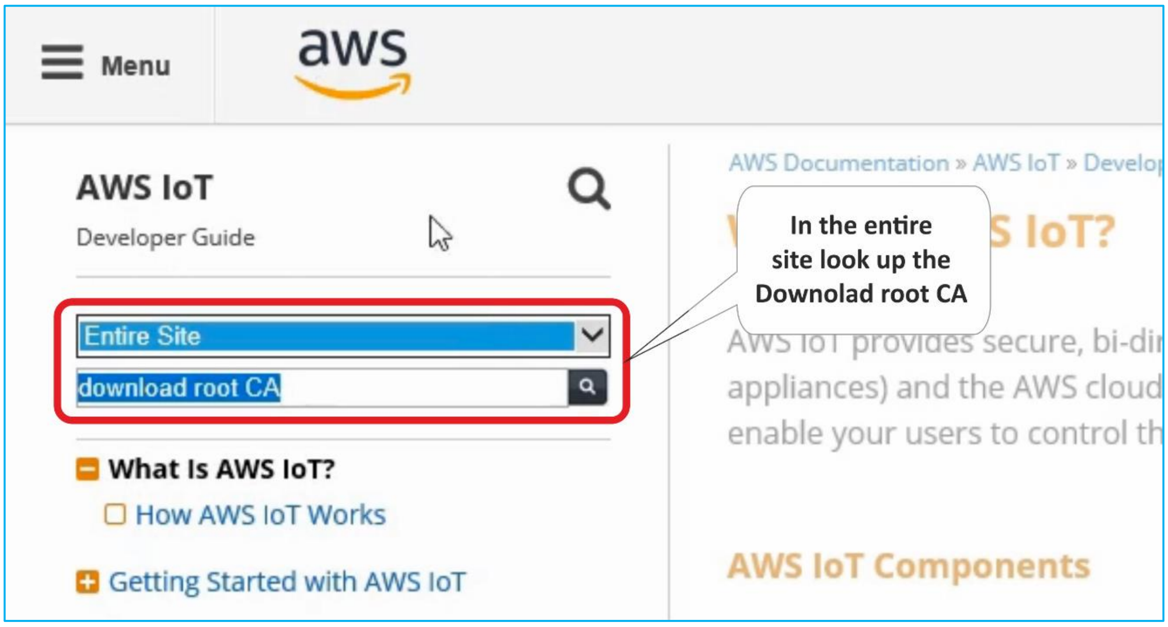
Explore other parts of AWS IoT

See all of your things
The thing registry is a catalog of all of your things.

Learn about policies
You create AWS IoT and IAM policies to give permissions to authenticated identities.

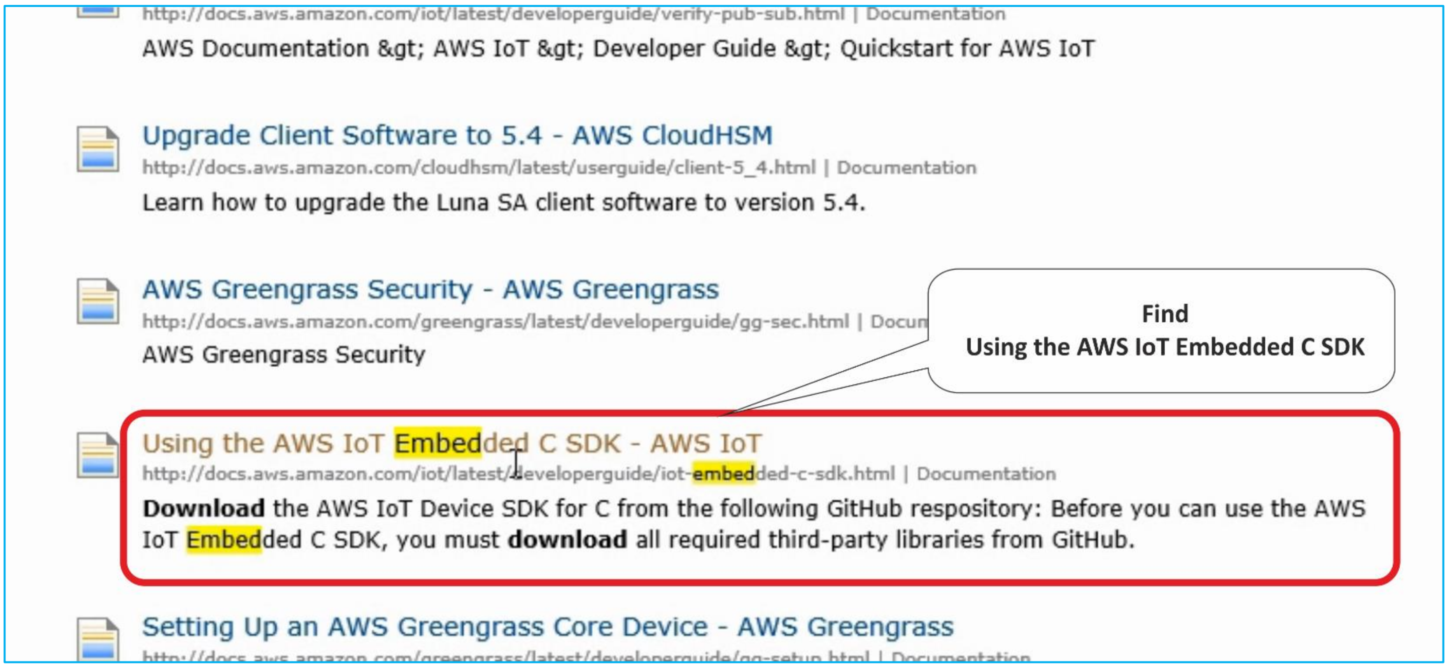
Dive into the documentation
Start exploring the documentation to get more details on the interworking of AWS IoT.

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


Menu
aws
AWS IoT
Developer Guide
Entire Site
download root CA
What Is AWS IoT?
How AWS IoT Works
Getting Started with AWS IoT
AWS Documentation » AWS IoT » Develop
In the entire site look up the Downolad root CA
AWS IoT Components

AWS IoT provides secure, bi-di
appliances) and the AWS cloud
enable your users to control th

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.



http://docs.aws.amazon.com/iot/latest/developerguide/verify-pub-sub.html | Documentation
AWS Documentation &gt; AWS IoT &gt; Developer Guide &gt; Quickstart for AWS IoT

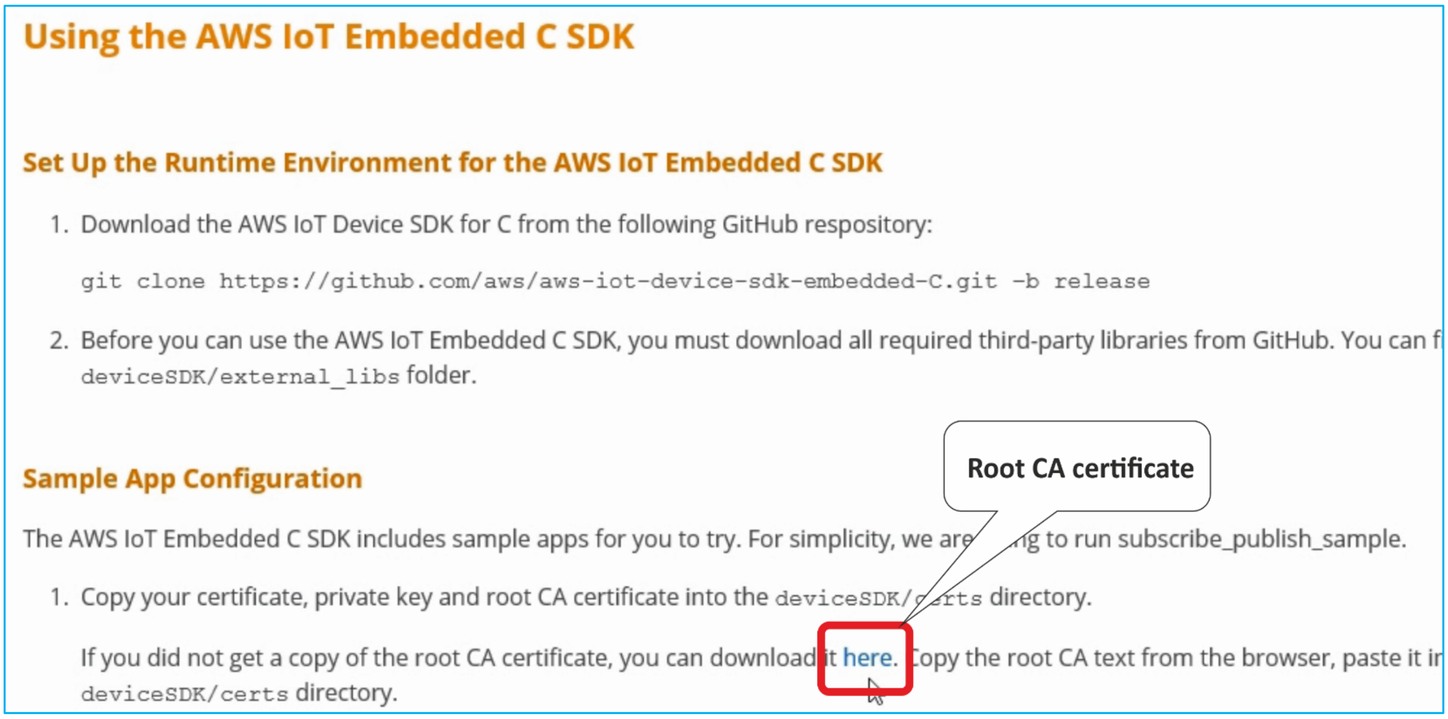
Upgrade Client Software to 5.4 - AWS CloudHSM
http://docs.aws.amazon.com/cloudhsm/latest/userguide/client-5_4.html | Documentation
Learn how to upgrade the Luna SA client software to version 5.4.

AWS Greengrass Security - AWS Greengrass
http://docs.aws.amazon.com/greengrass/latest/developerguide/gg-sec.html | Docum
AWS Greengrass Security

Using the AWS IoT Embedded C SDK - AWS IoT
http://docs.aws.amazon.com/iot/latest/developerguide/iot-embedded-c-sdk.html | Documentation
Download the AWS IoT Device SDK for C from the following GitHub respository: Before you can use the AWS IoT Embedded C SDK, you must **download** all required third-party libraries from GitHub.

Setting Up an AWS Greengrass Core Device - AWS Greengrass

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 respository:

   ```
   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 f
   `deviceSDK/external_libs` folder.

Sample App Configuration

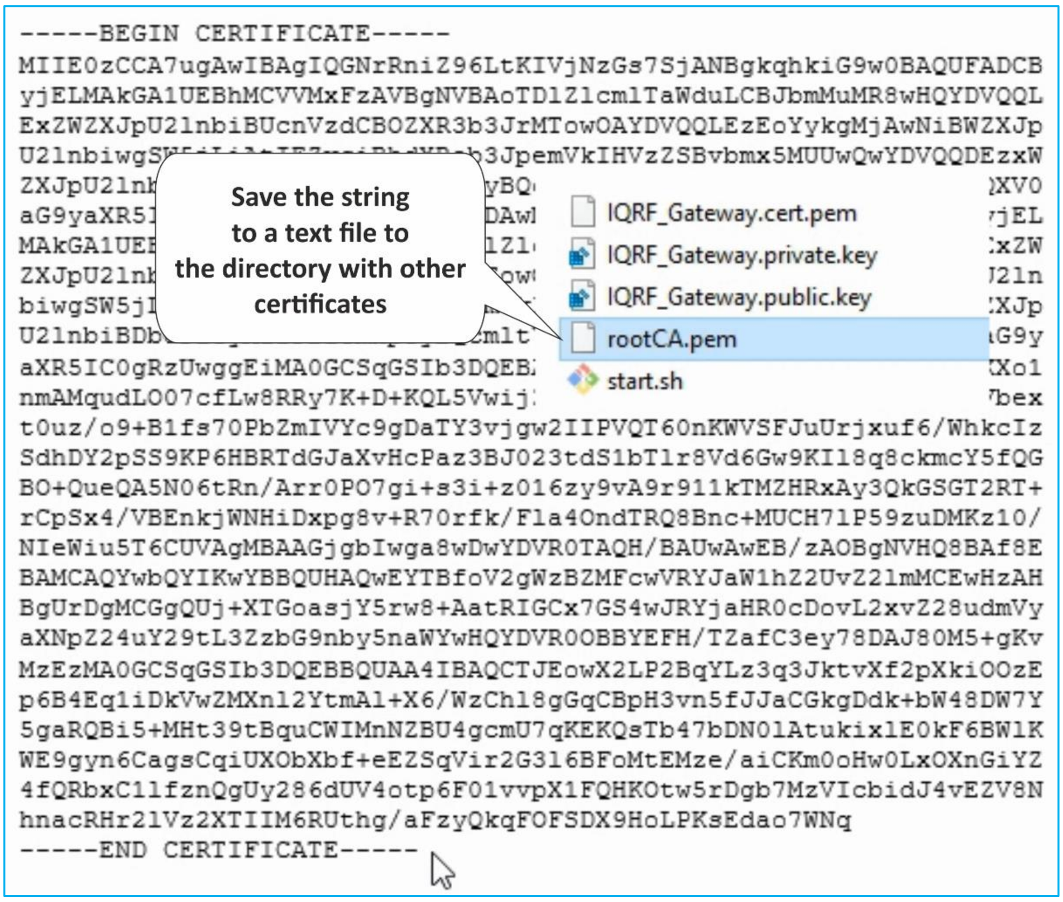
The AWS IoT Embedded C SDK includes sample apps for you to try. For simplicity, we are [illegible]g to run subscribe_publish_sample.

1. Copy your certificate, private key and root CA certificate into the `deviceSDK/c[illegible]rts` 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
   `deviceSDK/certs` directory.

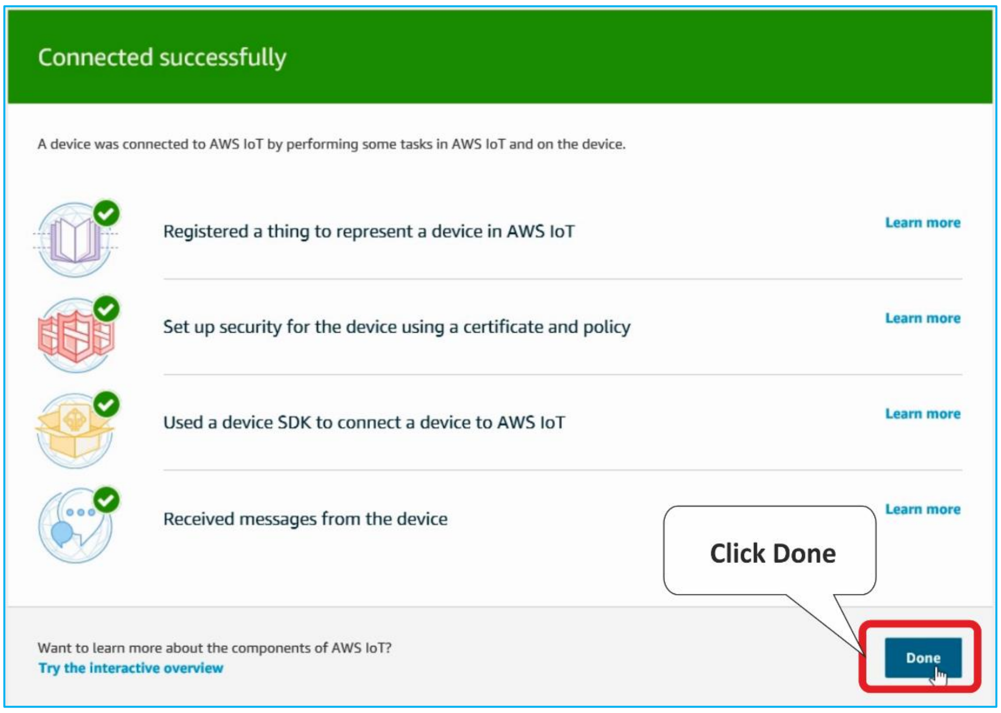
Copy the string to a text file and save it as **rootCA.pem** to the directory with other certificates.

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



```
-----BEGIN CERTIFICATE-----
MIIE0zCCA7ugAwIBAgIQGNrRniZ96LtKIVjNzGs7SjANBgkqhkiG9w0BAQUFADCB
yjELMAkGA1UEBhMCVVMxFzAVBgNVBAoTDlZlcmlTaWduLCBJbmMuMR8wHQYDVQQL
ExZWZXJpU2lnbiBUcnVzdCBOZXR3b3JrMTowOAYDVQQLEzEoYykgMjAwNiBWZXJp
U2lnbiwgS[illegible]b3JpemVkIHVzZSBvbmx5MUUwQwYDVQQDEzxW
ZXJpU2lnk[illegible]yBQ[illegible])XV0
aG9yaXR5I[illegible]DAwI[illegible]jEL
MAkGA1UEE[illegible]1Z1[illegible]xZW
ZXJpU2lnk[illegible]ow[illegible]J2ln
biwgSW5jI[illegible]XJp
U2lnbiBDb[illegible]zmlt[illegible]G9y
aXR5IC0gRzUwggEiMA0GCSqGSIb3DQEB[illegible]Xo1
nmAMqudLO07cfLw8RRy7K+D+KQL5Vwij[illegible]bex
t0uz/o9+B1fs70PbZmIVYc9gDaTY3vjgw2IIPVQT60nKWVSFJuUrjxuf6/WhkcIz
SdhDY2pSS9KP6HBRTdGJaXvHcPaz3BJ023tdS1bTlr8Vd6Gw9KIl8q8ckmcY5fQG
BO+QueQA5N06tRn/Arr0PO7gi+s3i+z016zy9vA9r911kTMZHRxAy3QkGSGT2RT+
rCpSx4/VBEnkjWNHiDxpg8v+R70rfk/Fla4OndTRQ8Bnc+MUCH7lP59zuDMKz10/
NIeWiu5T6CUVAgMBAAGjgbIwga8wDwYDVR0TAQH/BAUwAwEB/zAOBgNVHQ8BAf8E
BAMCAQYwbQYIKwYBBQUHAQwEYTBfoV2gWzBZMFcwVRYJaW1hZ2UvZ2lmMCEwHzAH
BgUrDgMCGgQUj+XTGoasjY5rw8+AatRIGCx7GS4wJRYjaHR0cDovL2xvZ28udmVy
aXNpZ24uY29tL3ZzbG9nby5naWYwHQYDVR0OBBYEFH/TZafC3ey78DAJ80M5+gKv
MzEzMA0GCSqGSIb3DQEBBQUAA4IBAQCTJEowX2LP2BqYLz3q3JktvXf2pXkiOOzE
p6B4Eq1iDkVwZMXnl2YtmAl+X6/WzChl8gGqCBpH3vn5fJJaCGkgDdk+bW48DW7Y
5gaRQBi5+MHt39tBquCWIMnNZBU4gcmU7qKEKQsTb47bDN0lAtukixlE0kF6BWlK
WE9gyn6CagsCqiUXObXbf+eEZSqVir2G3l6BFoMtEMze/aiCKm0oHw0LxOXnGiYZ
4fQRbxC1lfznQgUy286dUV4otp6F01vvpX1FQHKOtw5rDgb7MzVIcbidJ4vEZV8N
hnacRHr2lVz2XTIIM6RUthg/aFzyQkqFOFSDX9HoLPKsEdao7WNq
-----END CERTIFICATE-----
```

Next, click Done.


Connected successfully

A device was connected to AWS IoT by performing some tasks in AWS IoT and on the device.

Registered a thing to represent a device in AWS IoT

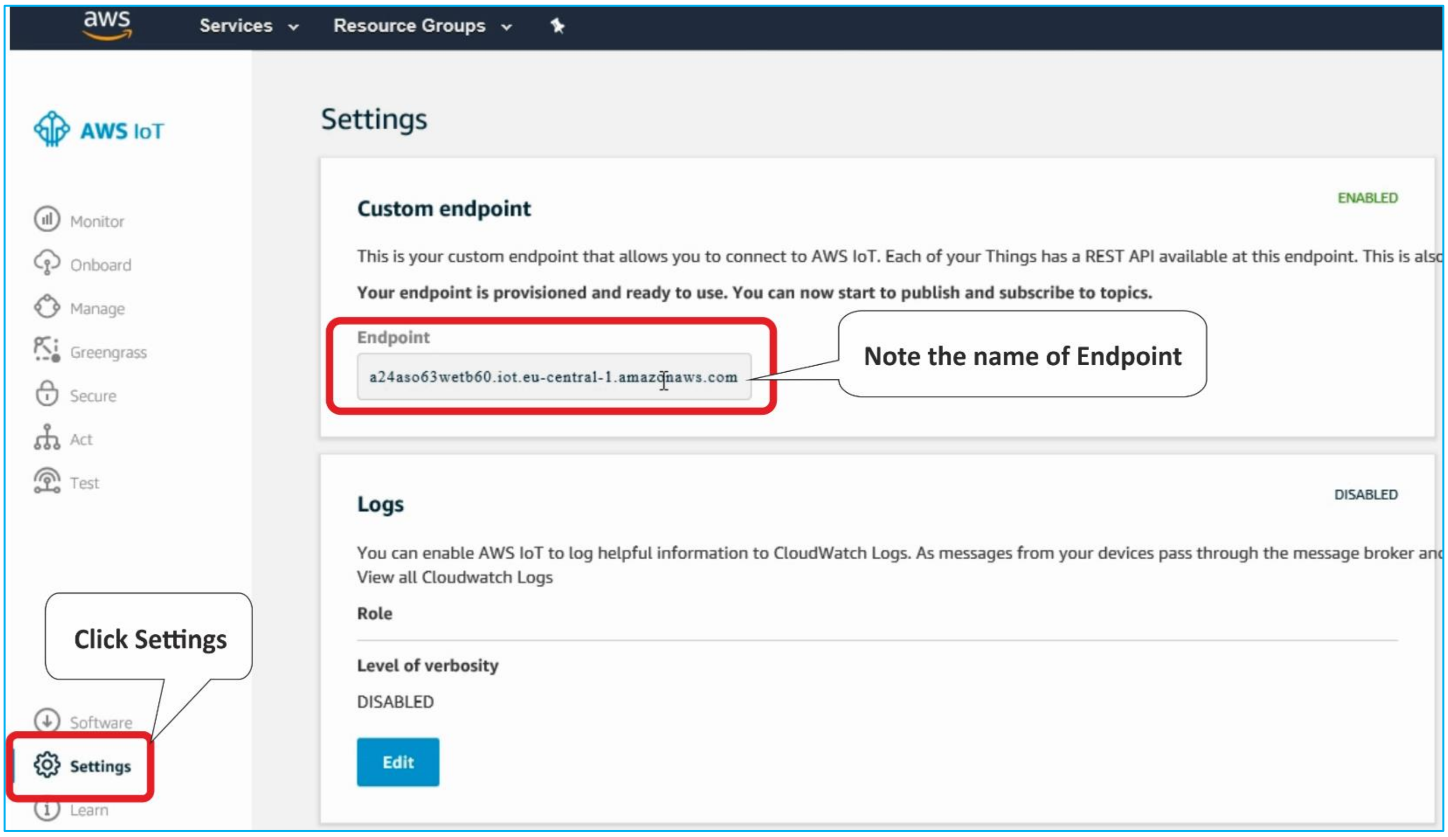
Set up security for the device using a certificate and policy

Used a device SDK to connect a device to AWS IoT

Received messages from the device

Want to learn more about the components of AWS IoT?
Try the interactive overview

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


Settings

Custom endpoint

ENABLED

This is your custom endpoint that allows you to connect to AWS IoT. Each of your Things has a REST API available at this endpoint. This is also

Your endpoint is provisioned and ready to use. You can now start to publish and subscribe to topics.

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

Logs

DISABLED

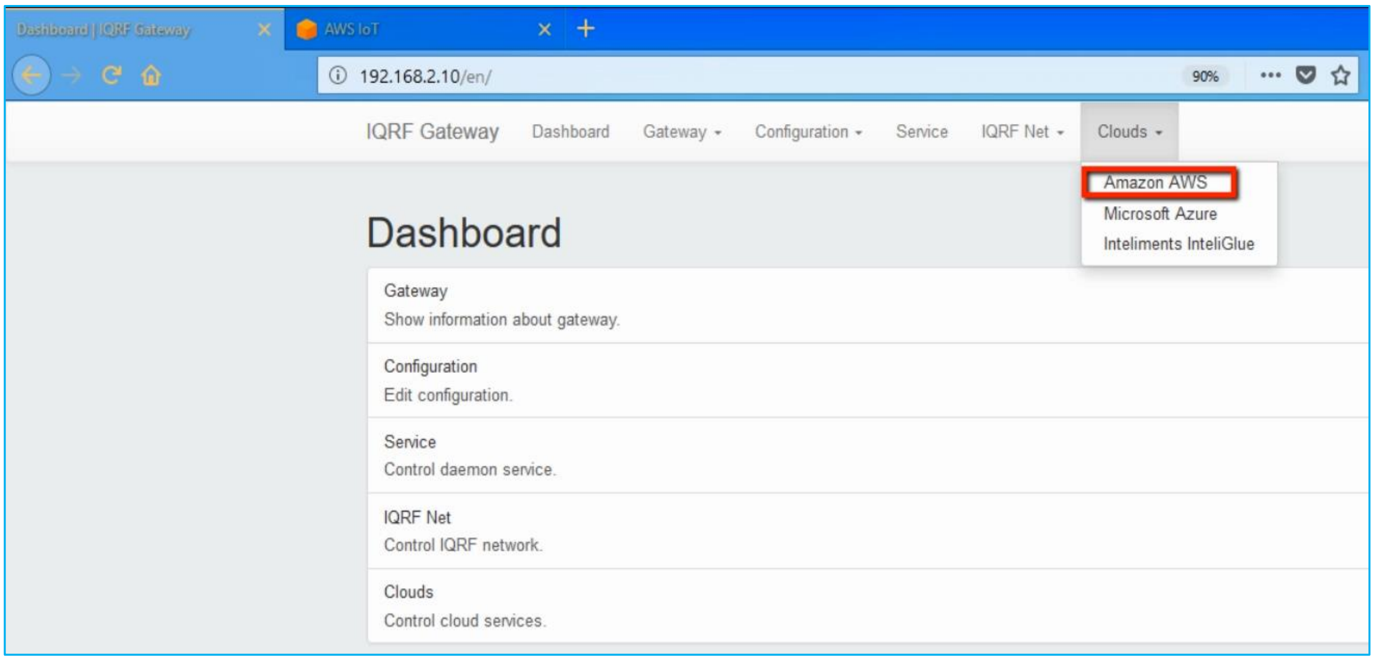
You can enable AWS IoT to log helpful information to CloudWatch Logs. As messages from your devices pass through the message broker and
View all Cloudwatch Logs

Role

Level of verbosity
DISABLED

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.

Go to the **IQRF Gateway Daemon web application** and in the **Clouds** menu click on the **Amazon AWS** item.



Dashboard

Gateway
Show information about gateway.

Configuration
Edit configuration.

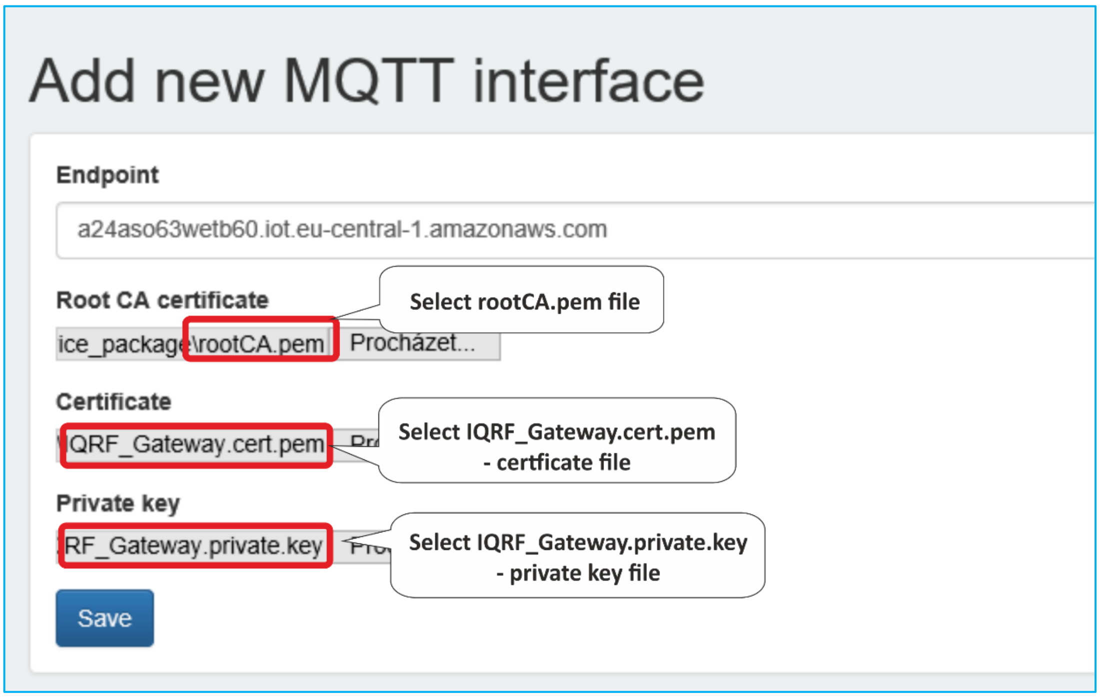
Service
Control daemon service.

IQRF Net
Control IQRF network.

Clouds
Control cloud services.

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

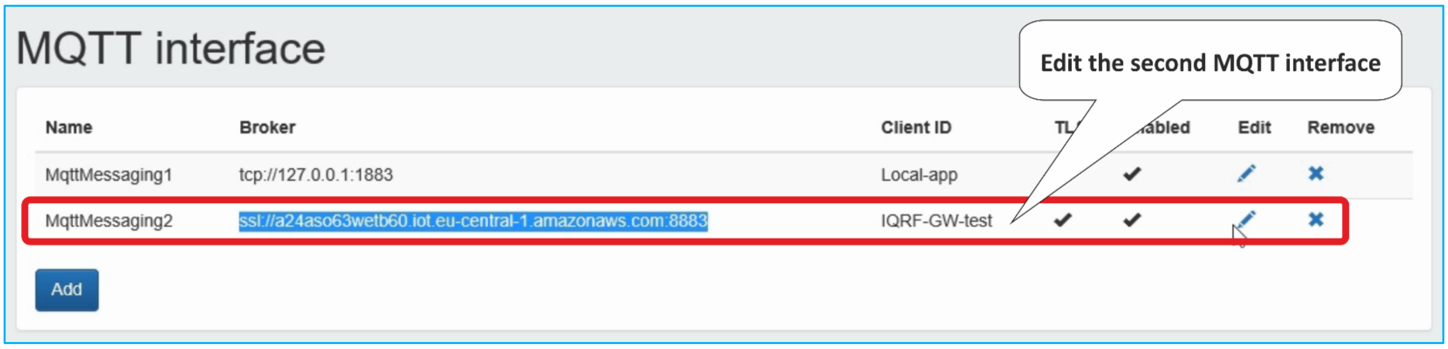
Root CA certificate: ice_package\rootCA.pem

Certificate: IQRF_Gateway.cert.pem

Private key: RF_Gateway.private.key

Procházet...
Select rootCA.pem file
Select IQRF_Gateway.cert.pem - certficate file
Select IQRF_Gateway.private.key - private key file
Save

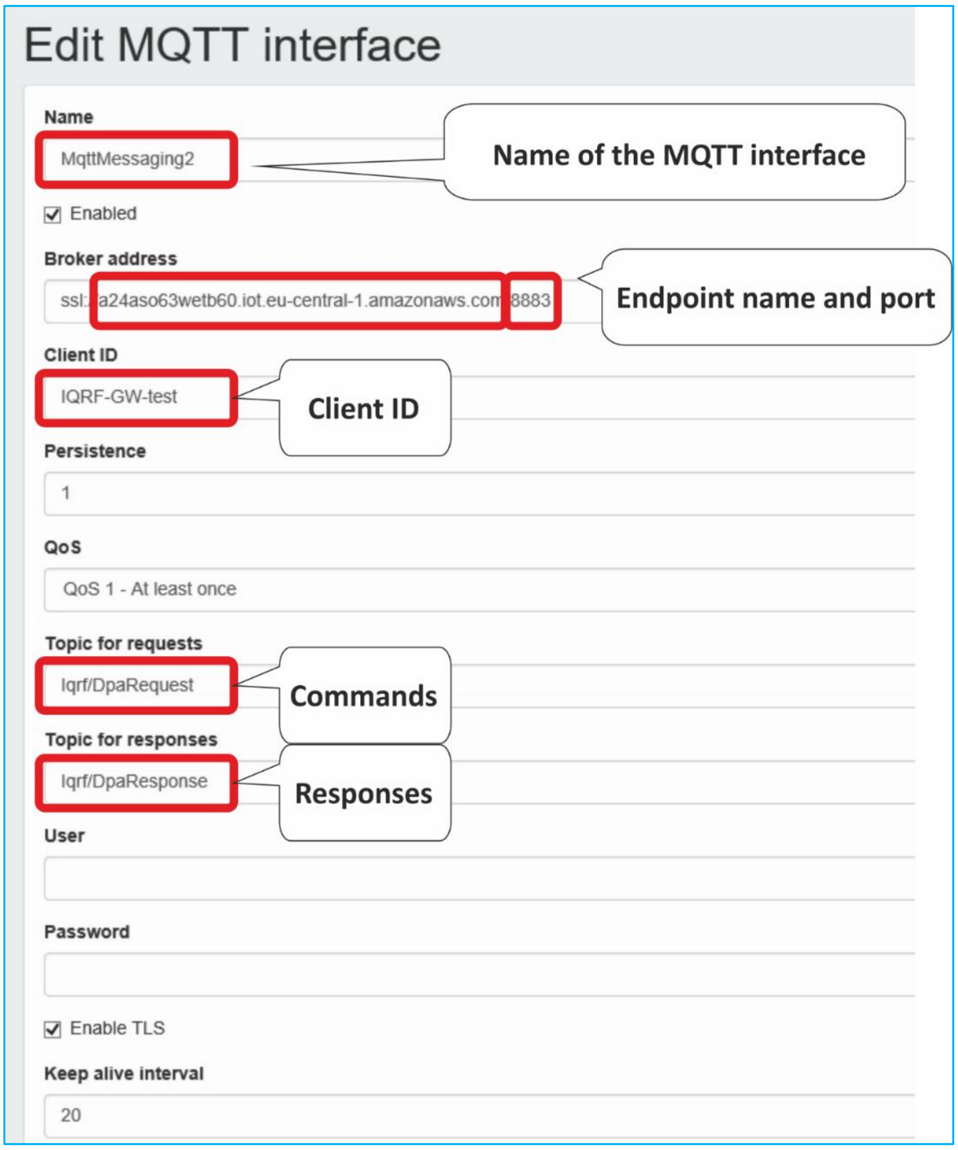
Inspect the new MQTT interface.



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	✔	✔	✎	✖

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

☑ Enabled

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

Client ID: IQRF-GW-test

Persistence: 1

QoS: QoS 1 - At least once

Topic for requests: Iqrf/DpaRequest

Topic for responses: Iqrf/DpaResponse

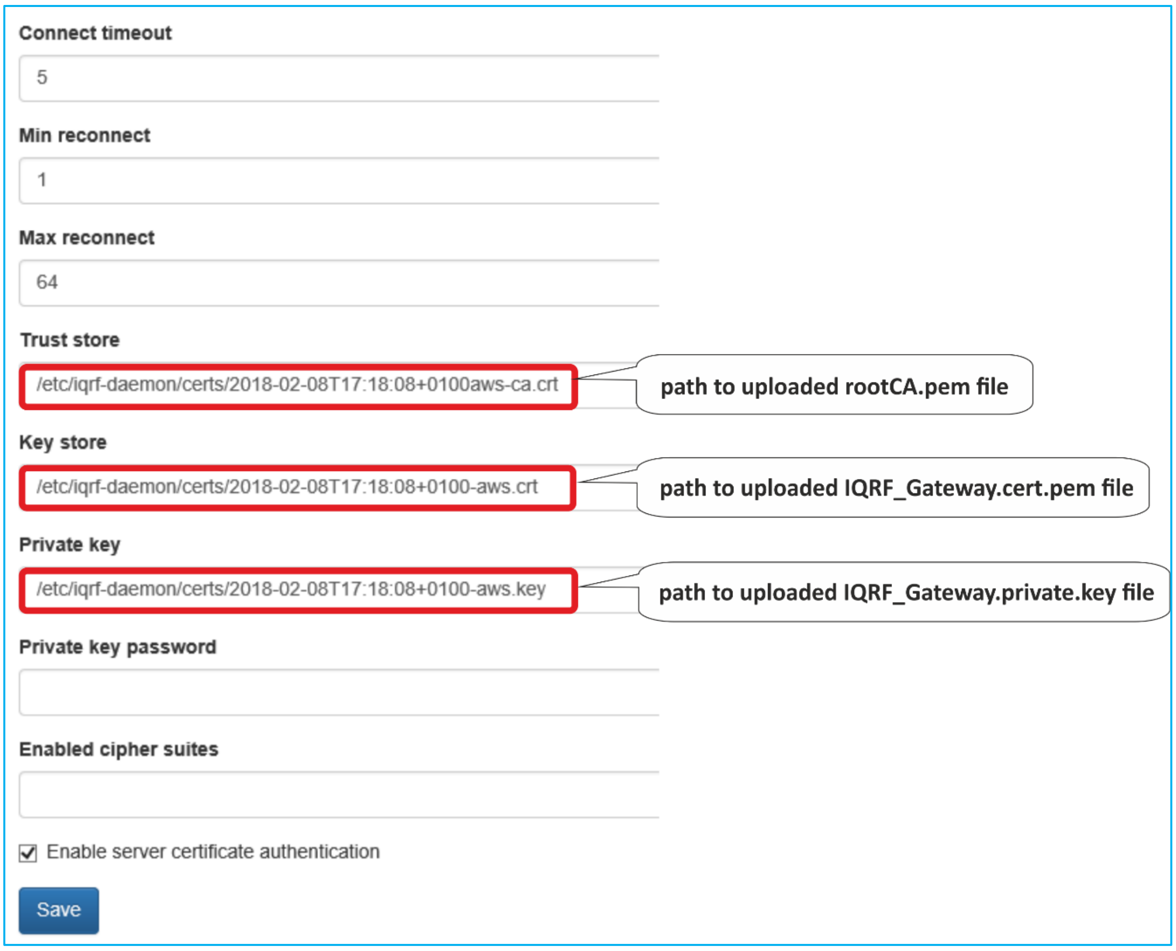
User:

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

Key store
/etc/iqrf-daemon/certs/2018-02-08T17:18:08+0100-aws.crt

Private key
/etc/iqrf-daemon/certs/2018-02-08T17:18:08+0100-aws.key

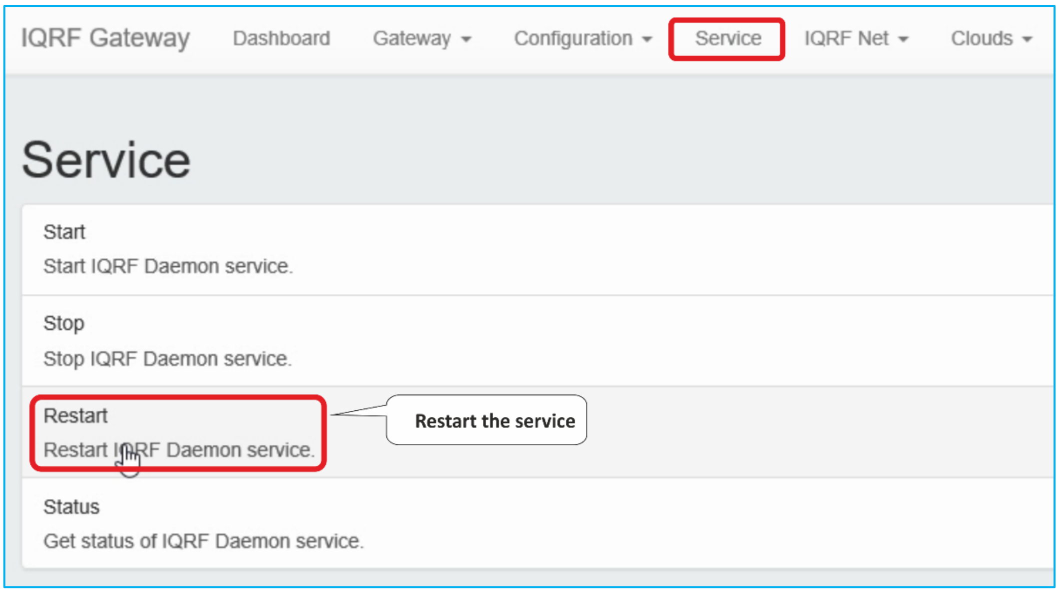
Private key password

Enabled cipher suites

☑ Enable server certificate authentication

path to uploaded rootCA.pem file
path to uploaded IQRF_Gateway.cert.pem file
path to uploaded IQRF_Gateway.private.key file
Save

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



Service

Start
Start IQRF Daemon service.

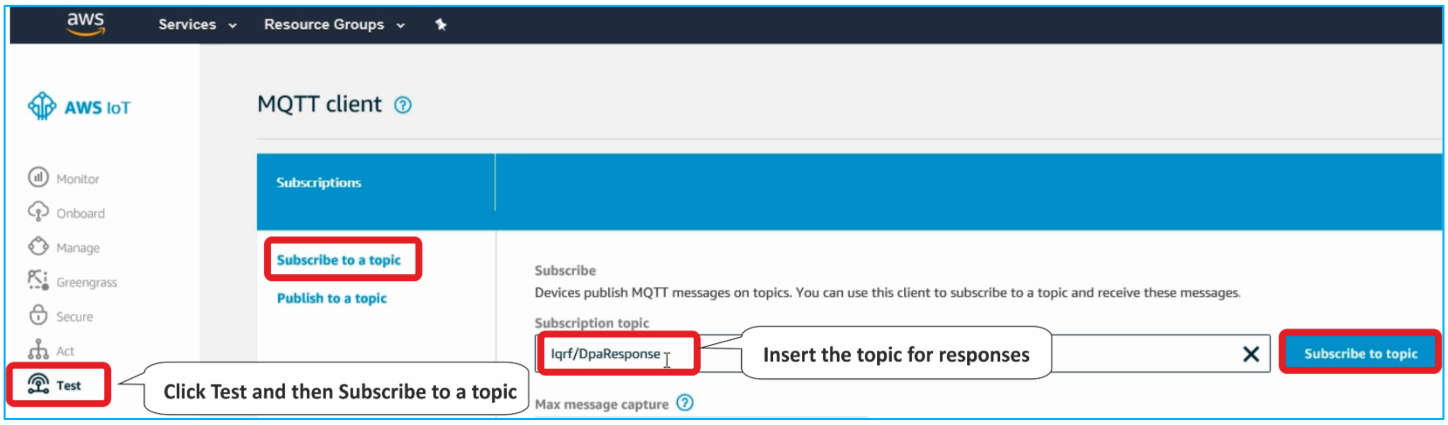
Stop
Stop IQRF Daemon service.

Restart
Restart IQRF Daemon service.

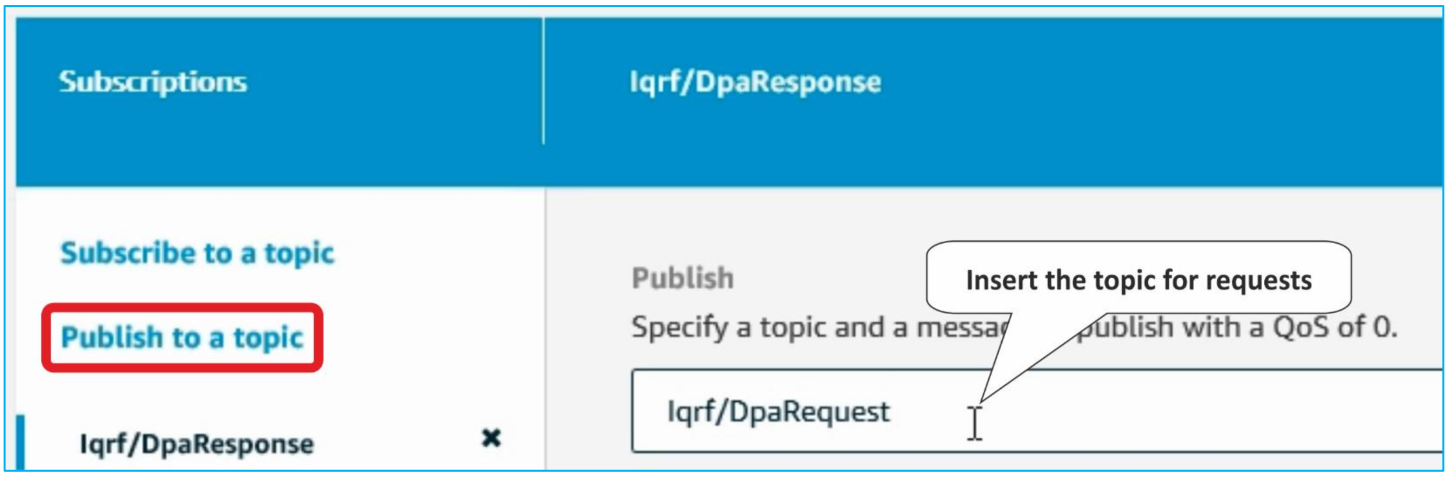
Status
Get status of IQRF Daemon service.

Test the connection

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



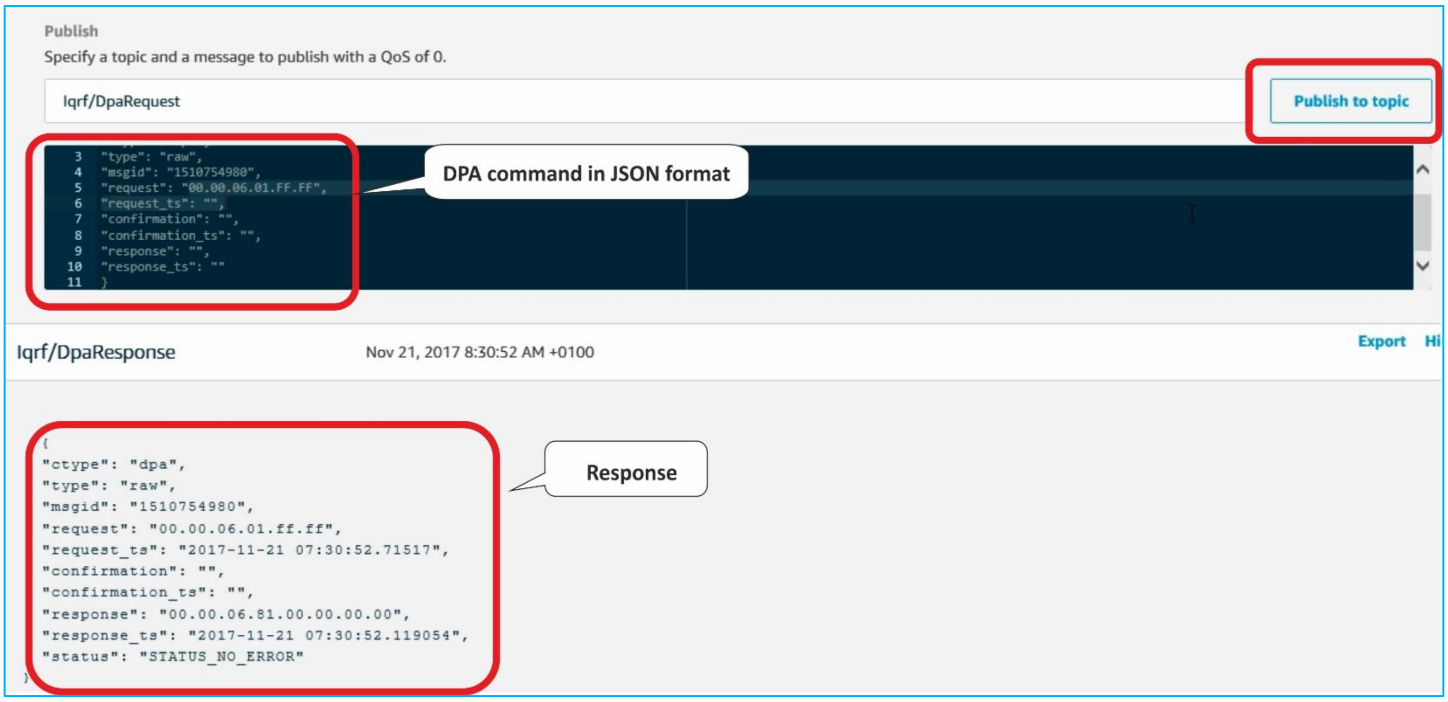
To send commands from the cloud to the gateway, set the **Iqrf/DpaRequest** as the topic for requests. Gateway will expect commands in this topic.



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": ""
}
```

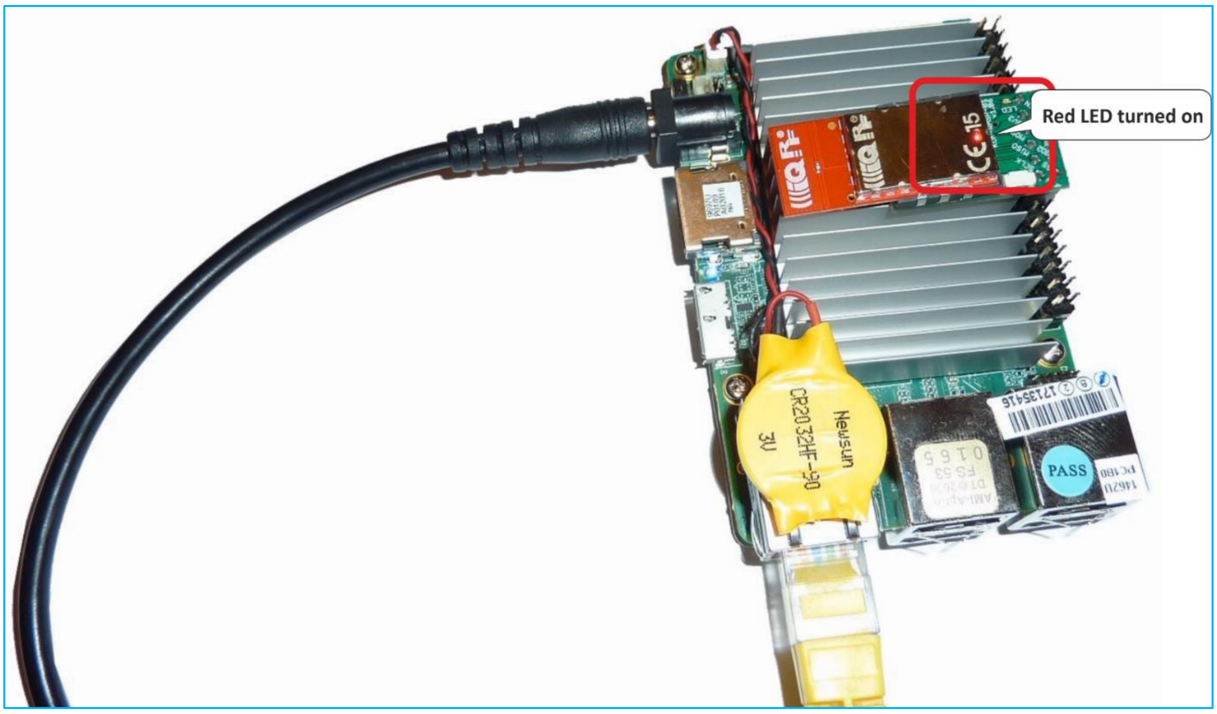
We can see that the gateway picked up and executed the command, and sent a confirmation with "No Error" into the **Iqrf/DpaResponse** topic.


Publish
Specify a topic and a message to publish with a QoS of 0.
Iqrf/DpaRequest
Publish to topic
DPA command in JSON format
Iqrf/DpaResponse
Nov 21, 2017 8:30:52 AM +0100
Export
Response

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

```
{
"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"
}
```

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.