RAK7249 Supported LoRa Network Servers

Amazon Web Services (AWS)

Execute the following steps to set up your AWS account and permissions:

Set up Roles and Policies in IAM

Add an IAM Role for CUPS Server

Adding an IAM role will allow the Configuration and Update Server (CUPS) to handle the wireless gateway credentials.

This procedure needs to be done only once, but must be performed before a LoRaWAN gateway tries to connect with AWS IoT Core for LoRaWAN.

1. Go to the IAM Roles page on the IAM console.
2. Choose **Create role**.
3. On the Create Role page, choose **Another AWS account**.
4. Enter your **Account ID**, then select **Next: Permissions**.
5. In the search box next to the Filter Policies, type ***AWSIoTWirelessGatewayCertManager***.
   - If the search results show the policy named ***AWSIoTWirelessGatewayCertManager***, select it by clicking the checkbox.
   - If the policy does not exist, create one.
     - Go to the IAM console .
     - Choose **Policies** from the navigation pane.
     - Choose **Create Policy**, then select the **JSON** tab to open the policy editor.
     - Replace the existing template with trust policy document.

```json
{
"Version": "2012-10-17",
"Statement": [
{
"Sid": "IoTWirelessGatewayCertManager",
"Effect": "Allow",
"Action": [
"iot:CreateKeysAndCertificate",
"iot:DescribeCertificate",
"iot:ListCertificates",
"iot:RegisterCertificate"
        ],
"Resource": "*"
         }
     ]
}
```

     - Choose **Review Policy** to open the Review Page.
     - For the Name, type ***AWSIoTWirelessGatewayCertManager***.

> 📝 **NOTE:**
>
> You must enter the name as ***AWSIoTWirelessGatewayCertManager*** and must not use a different name. This is for consistency with future releases.

- For the Description, enter a description of your choice.
- Then choose **Create policy**. You will see a confirmation message showing the policy has been created.

6. Choose **Next: Tags**, then **Next: Review**.
7. In Role name, enter ***IoTWirelessGatewayCertManagerRole***, and then choose to **Create role**.

> 📝 **NOTE:**
>
> You must not use a different name. This is for consistency with future releases.

8. In the confirmation message, choose ***IoTWirelessGatewayCertManagerRole*** to edit the new role.
9. In the **Summary**, choose the **Trust relationships** tab, and then choose **Edit trust relationship**.
10. In the **Policy Document**, change the **Principal** property to represent the IoT Wireless service:

```json
"Principal": {
"Service": "iotwireless.amazonaws.com"
},
```

- After changing the Principal property, the complete policy document should look like the following:

```json
{
"Version": "2012-10-17",
"Statement": [
    {
"Effect": "Allow",
"Principal": {
"Service": "iotwireless.amazonaws.com"
       },
"Action": "sts:AssumeRole",
"Condition": {}
    }
    ]
}
```

11. Choose **Update Trust Policy** to save your changes and exit. At this point, you have created the ***IoTWirelessGatewayCertManagerRole*** and you won't need to do this again.

> 📝 **NOTE:**
>
> The examples in this document are intended only for dev environments. All devices in your fleet must have credentials with privileges that authorize only intended actions on specific resources. The specific permission policies can vary for your use case. Identify the permission policies that best meet your business and security requirements. For more information, refer to **Example Policies** and **Security Best Practices**

Add IAM Role for Destination to AWS IoT Core for LoRaWAN

Creating a Policy

Creating a policy gives the role permissions to describe the IoT endpoint and publish messages to AWS IoT.

1. Go to the IAM console .
2. Choose **Policies** from the navigation pane.
3. Choose **Create Policy**, then choose the **JSON** tab to open the policy editor. Replace the existing template with this trust policy document:

```json
{
"Version": "2012-10-17",
"Statement": [
   {
"Effect": "Allow",
"Action":
[
"iot:DescribeEndpoint",
"iot:Publish"
],
"Resource": "*"
   }
   ]
}
```

4. Choose **Review Policy** to open the Review page.
5. For **Name**, enter a name of your choice.
6. For **Description**, enter a description of your choice.
7. Choose **Create policy**. You will see a confirmation message indicating that the policy has been created.

Creating the Role

1. In the **IAM console**, choose **Roles** from the navigation pane to open the Roles page.
2. Choose **Create Role**.
3. In **Select type of trusted entity**, choose **Another AWS account**.
4. In **Account ID**, enter your AWS account ID, and then choose **Next: Permissions**.
5. Search for the **IAM policy** you just created by entering the policy name in the search bar.
6. In the search results, select the checkbox corresponding to the policy.
7. Choose **Next: Tags**.
8. Choose **Next: Review** to open the Review page.
9. For **Role name**, enter an appropriate name of your choice.
10. For **Description**, enter a description of your choice.
11. Choose **Create role**. You will see a confirmation message indicating that your role has been created.

Updating your Trust Policy

Update your role's trust relationship to grant AWS IoT Core for LoRaWAN permission to assume this IAM role when delivering messages from devices to your account.

1. In the IAM console, choose **Roles** from the navigation pane to open the Roles page.
2. Enter the name of the role you created earlier in the search window, and click on the role name in the search results. This opens up the Summary page.
3. Choose the **Trust relationships table** to navigate to the Trust relationships page.

4. Choose **Edit trust relationship**. The principal AWS role in your trust policy document defaults to root and must be changed. Replace the existing policy with this:

```json
{
"Version": "2012-10-17",
"Statement": [
{
"Sid": "",
"Effect": "Allow",
"Principal": {
"Service": "iotwireless.amazonaws.com"
},
"Action": "sts:AssumeRole",
"Condition": {}
   }
   ]
}
```

5. Choose **Update Trust Policy.** Under Trusted entities, you will see: *The identity provider(s) iotwireless.amazonaws.com*.

Add the Gateway to AWS IoT

Requirements

To complete setting up your gateway, you need the following:

- LoRaWAN region. For example, if the gateway is deployed in a US region, the gateway must support LoRaWAN region US915.
- Gateway LNS-protocols. Currently, the LoRa Basics Station protocol is supported.
- Gateway ID (GatewayEUI) or serial number. This is used to establish the connection between the LNS and the gateway. Consult the documentation for your gateway to locate this value.
- Add minimum software versions required, including Basics Station 2.0.5.

Add the LoRaWAN Gateway

To register the Gateway with AWS IoT Core for LoRaWAN, execute these steps:

1. Go to the AWS IoT console .
2. Select **Wireless connectivity** in the navigation panel on the left.
3. Choose **Intro**, and then choose **Get started**. This step is needed to pre-populate the default profiles.
4. Under **Add LoRaWAN gateways and wireless devices**, choose **Add gateway**.
5. In the **Add gateway section**, fill in the **GatewayEUI** and **Frequency band (RF Region)** fields.
6. Enter a descriptive name in the **Name** – optional field. It is recommended that you use the GatewayEUI as the name.
7. Choose **Add gateway**.
8. On the **Configure your Gateway** page, find the section titled **Gateway certificate**.
9. Select **Create certificate**.
10. Once the **Certificate created and associated with your gateway** message is shown, select **Download certificates** to download the certificate (*xxxxx.cert.pem*) and private key (*xxxxxx.private.key*).
11. In the section **Provisioning credentials**, choose **Download server trust certificates** to download the **CUPS (cups.trust)** and **LNS (lns.trust)** server trust certificates.
12. Copy the CUPS and LNS endpoints and save them for use while configuring the gateway.
13. Choose **Submit** to add the gateway.

Add a LoRaWAN Device to AWS IoT

Requirements:

- Locate and note the following specifications about your endpoint device.
  - **LoRaWAN Region**: This must match the gateway LoRaWAN region. The following Frequency bands (RF regions) are supported: o EU868 o US915 o EU433
  - **MAC Version**: This must be one of the following: o V1.0.2 o v1.0.3 o v1.1
  - OTAA v1.0x and OTAA v1.1 are supported.
  - ABP v1.0x and ABP v1.1 are supported.
- Locate and note the following information from your device manufacturer:
  - For OTAA v1.0x devices: DevEUI, AppKey, AppEUI
  - For OTAA v1.1 devices: DevEUI, AppKey, NwkKey, JoinEUI
  - For ABP v1.0x devices: DevEUI, DevAddr, NwkSkey, AppSkey
  - For ABP v1.1 devices: DevEUI, DevAddr, NwkSEnckey, FNwkSIntKey, SNwkSIntKey, AppSKey

Verify Profiles

AWS IoT Core for LoRaWAN supports device profiles and service profiles. Device profiles contain the communication and protocol parameter values the device needs to communicate with the network server. Service profiles describe the communication parameters the device needs to communicate with the application server.

Some pre-defined profiles are available for device and service profiles. Before proceeding, verify that these profile settings match the devices you will be setting up to work with AWS IoT Core for LoRaWAN.

1. Navigate to the AWS IoT console . In the navigation pane, choose **Wireless connectivity**.
2. In the navigation pane, choose **Profiles**.
3. In the **Device Profiles** section, there are some pre-defined profiles listed.
4. Check each of the profiles to determine if one of them will work for you.
5. If not, select **Add device profile** and set up the parameters as needed. For US 915 as an example, the values are:
   - MacVersion 1.0.3
   - RegParamsRevision RP002-1.0.1
   - MaxEirp 10
   - MaxDutyCycle 10
   - RfRegion US915
   - SupportsJoin true
6. Continue once you have a device profile that will work for you.
7. In the **Service Profiles** section, there are some pre-defined profiles listed. Check each of the profiles to determine if one of them will work for you.
8. If not, select **Add service profile** and set up the parameters as needed. As an example, the default service profile parameters are shown below. However, only the ***AddGwMetadata*** setting can be changed at this time.
   - UlRate 60
   - UlBucketSize 4096
   - DlRate 60

- DlBucketSize 4096
- AddGwMetadata true
- DevStatusReqFreq 24
- DrMax 15
- TargetPer 5
- MinGwDiversity 1

9. Proceed only if you have a device and service profile that will work for you.

Set up a Destination for Device Traffic

Because most LoRaWAN devices don't send data to AWS IoT Core for LoRaWAN in a format that can be consumed by AWS services, traffic must first be sent to a Destination. A Destination represents the AWS IoT rule that processes a device's data for use by AWS services. This AWS IoT rule contains the SQL statement that selects the device's data and the topic rule actions that send the result of the SQL statement to the services that will use it.

For more information on Destinations, refer to the AWS LoRaWAN Developer Guide .

A destination consists of a Rule and a Role. To set up the destination, execute the following steps:

1. Navigate to the AWS IoT console . In the navigation pane, choose **Wireless connectivity**, and then **Destinations**.
2. Choose **Add Destination**.
3. On the Add destination page, in the **Permissions** section, select the IAM role you had created earlier, from the drop-down.
4. Under **Destination details**, enter ***ProcessLoRa*** as the Destination name, and an appropriate description under **Destination description – optional**.

> 📝 **NOTE:**
>
> The Destination name can be anything. For getting started and consistency, choose ProcessLoRa for the first integration with AWS IoT Core for LoRaWAN.

5. For **Rule name**, enter ***LoRaWANRouting***. Ignore the section **Rules configuration – Optional** for now. The Rule will be set up later in the "Hello World" sample application. See Create the IoT Rule for the destination.
6. Choose **Add Destination**. You will see a message "*Destination added*", indicating the destination has been successfully added.

Register the Device

Now, register an endpoint device with AWS IoT Core for LoRaWAN as follows:

1. Go to the AWS IoT console .
2. Select **Wireless connectivity** in the navigation panel on the left.
3. Select **Devices**, then choose **Add wireless device**.
4. On the **Add device** page, select the LoRaWAN specification version in the drop-down under **Wireless device specification**.
5. Under **LoRaWAN specification and wireless device configuration**, enter the **DevEUI** and confirm it in the **Confirm DevEUI** field.
6. Enter the remaining fields as per the OTAA/ABP choice you made above.
7. Enter a name for your device in the **Wireless device name – optional field**.
8. In the **Profiles** section, under **Wireless device profile**, find a drop-down option that corresponds to your device and region.

> 📝 **NOTE:**
>
> Compare your device details to ensure the device profile is correct. If there are no valid default options, you will have to create a new profile. See the Verify Profiles section.

9. Choose **Next**.
10. Choose the destination you created earlier (*ProcessLoRa*) from the drop-down under **Choose destination**.
11. Choose **Add device**.
12. You will see a message saying "*Wireless device added*", indicating that your device has been set up successfully.

Set up the Gateway

- **Set up the Gateway Hardware**
- **Set up the Gateway Software**

Configure the Gateway Device

1. Using your preferred Web browser, input the aforementioned IP Address and you should see the same Log-in Page shown in the following image. Login the credentials provided below:

- Username: **root**
- Password: **root**

Figure 1: Web User Interface Log-in

2. The firmware version 1.2.0065_Release_r209 on the gateway supports AWS IoT Core for LoRaWAN, and it can be verified on **Status** > **Overview** > **System** > **Firmware Version**.

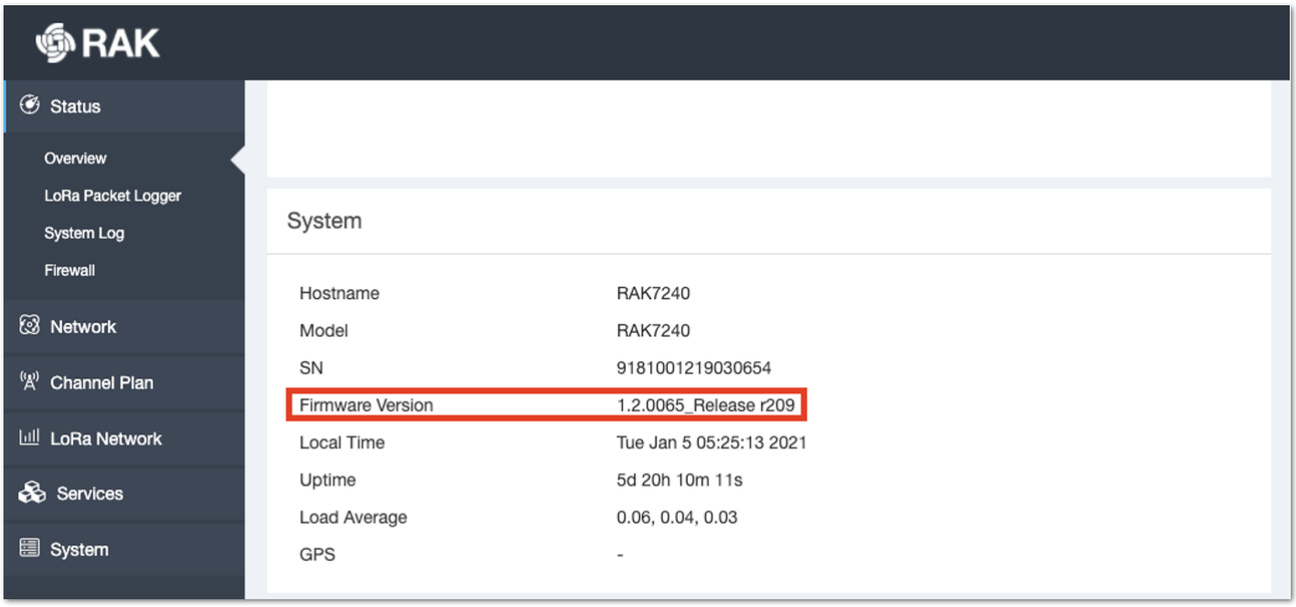

Figure 2: Checking the Firmware Version

3. If the firmware version is prior to 1.2.0065_Release_r209, upgrade the firmware. Navigate to **System** > **Backup/Flash Firmware** > **Flash new firmware image** > **Upgrade the firmware**.

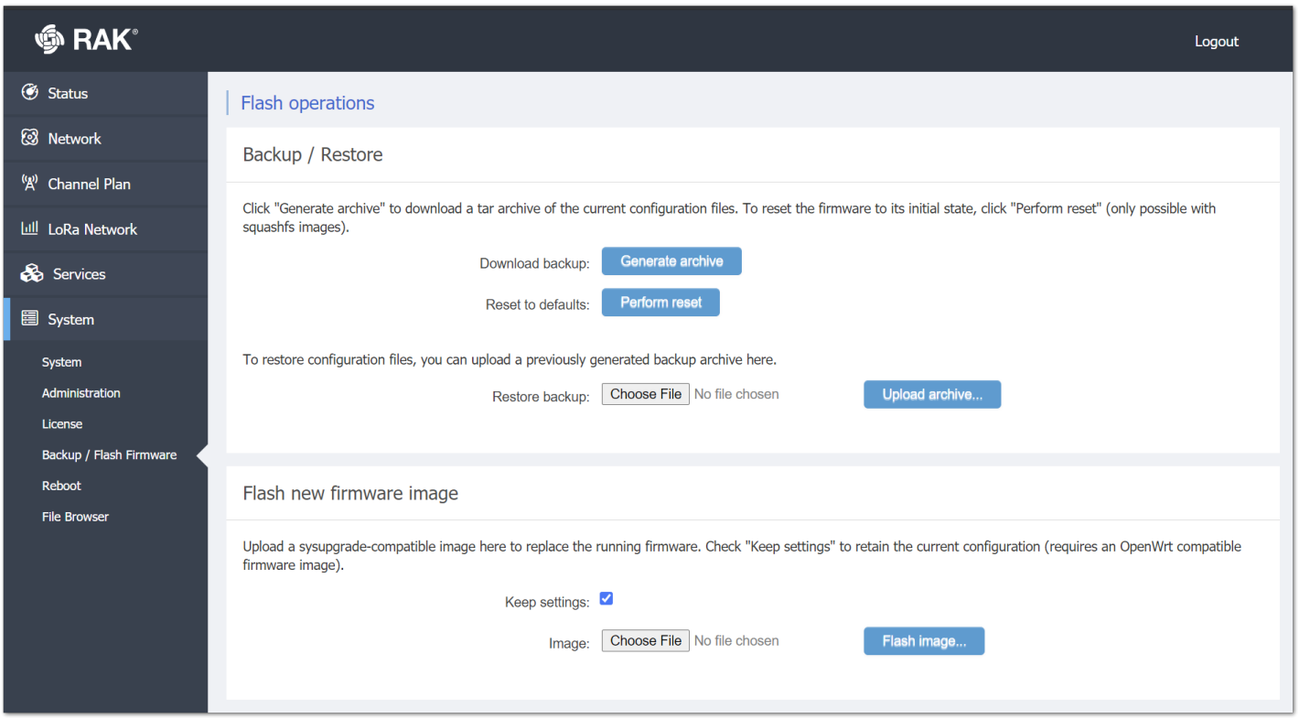

Figure 3: Flashing the firmware

4. Configure Network Mode to Basic Station. Navigate to **LoRa Network** then **Network Settings**.
   - Change the Mode in LoRaWAN Network Settings to Basic Station.
   - Select **LNS Server** from Server, then choose **TLS Server and Client Authentication** from Authentication Mode.

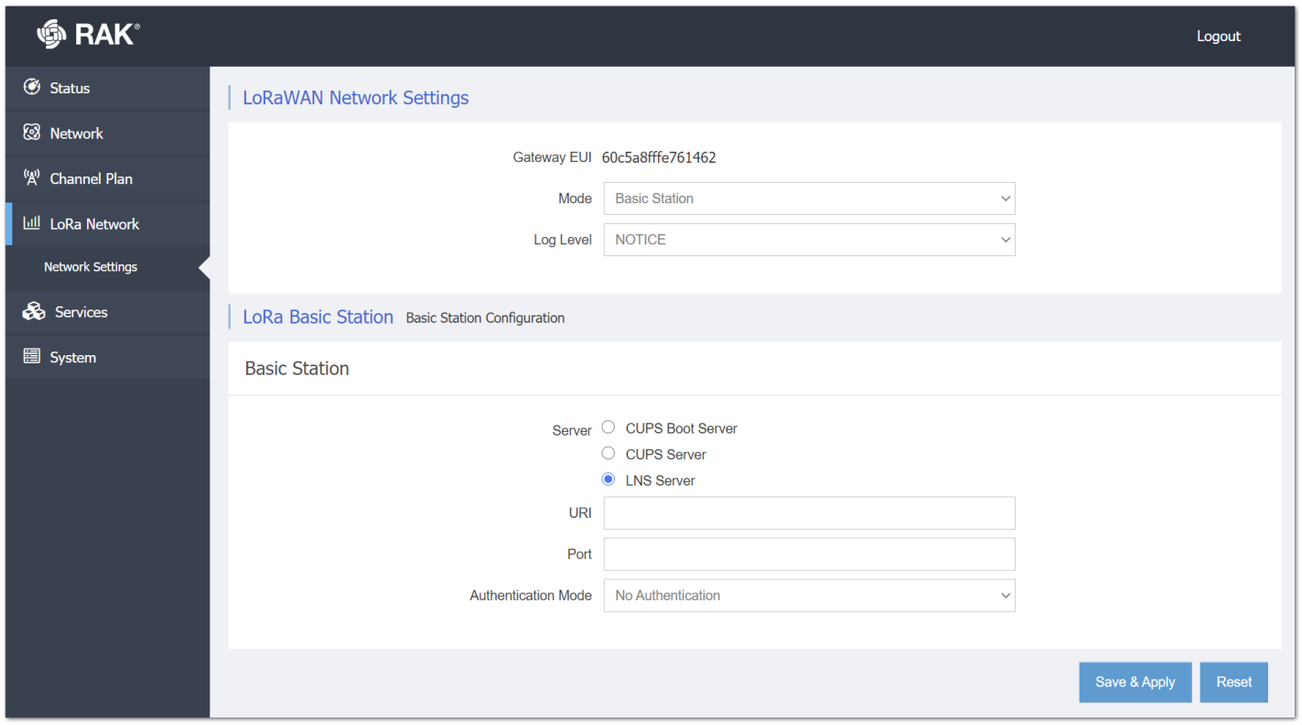

Figure 4: Configuring Network Mode to Basic Station

5. Configure URI, Port, and Authentication Mode.

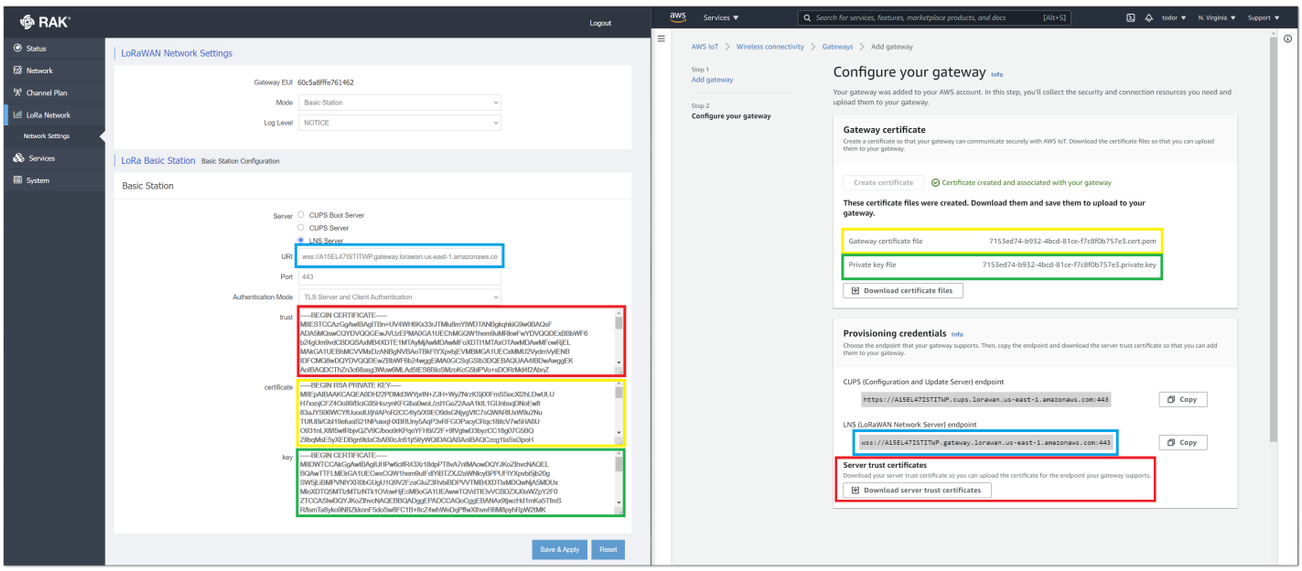

Figure 5: Configuring URI, Port, and Authentication Mode

6. Verifying Operation. Check if the gateway is online in AWS IoT console.

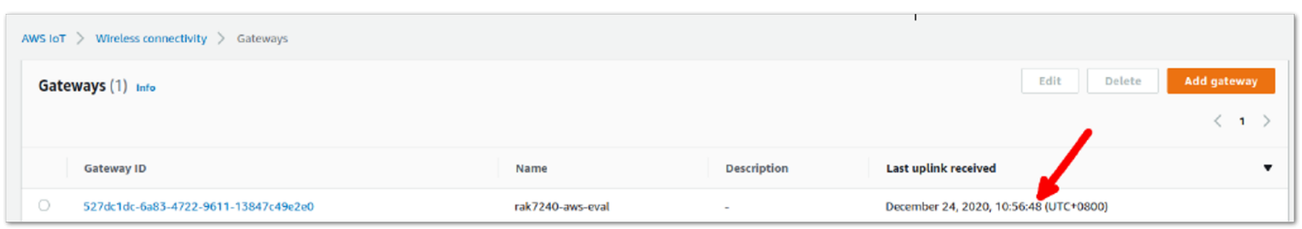

Figure 6: Verifying Operation

Add End Devices

This section shows an example of how to join the AWS IoT LoRaWAN server.

1. Add Device Profile.

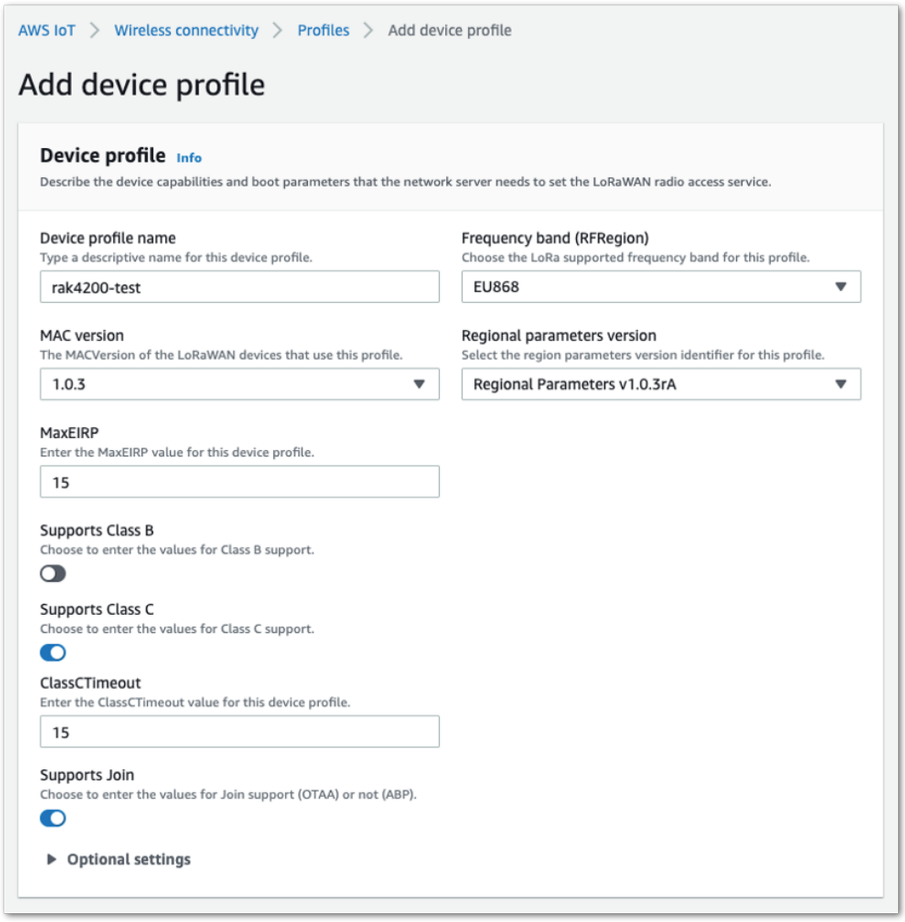

Figure 7: Adding the Device Profile

2. Add Service Profile.

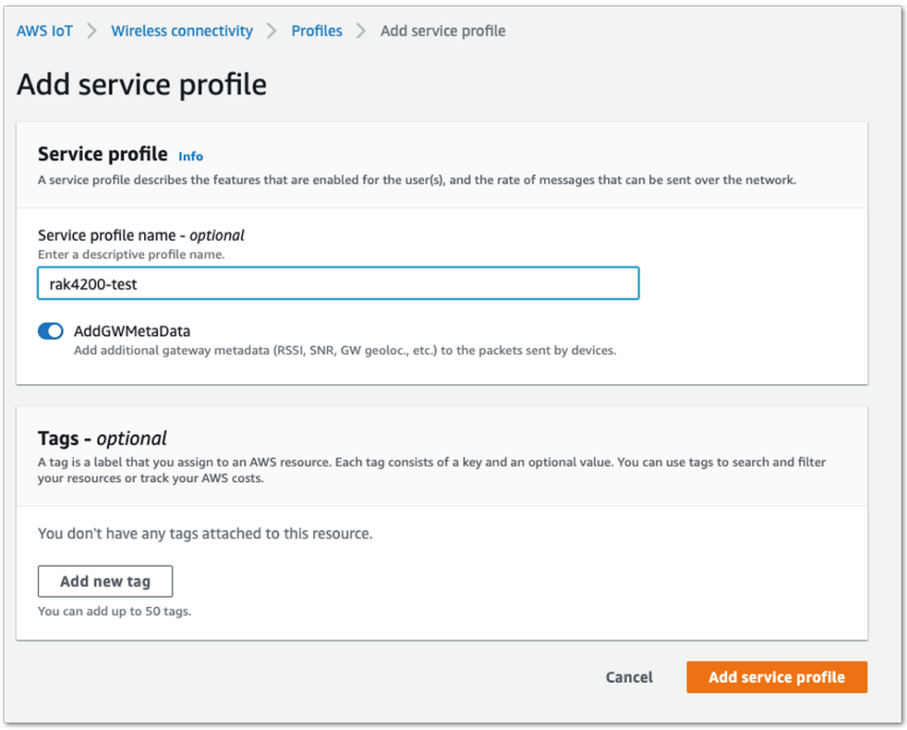

Figure 8: Adding the Service Profile

3. Add Destination.

Before adding the destination, follow the Add IAM role for Destination to AWS IoT Core for LoRaWAN section to configure IAM policy and role.

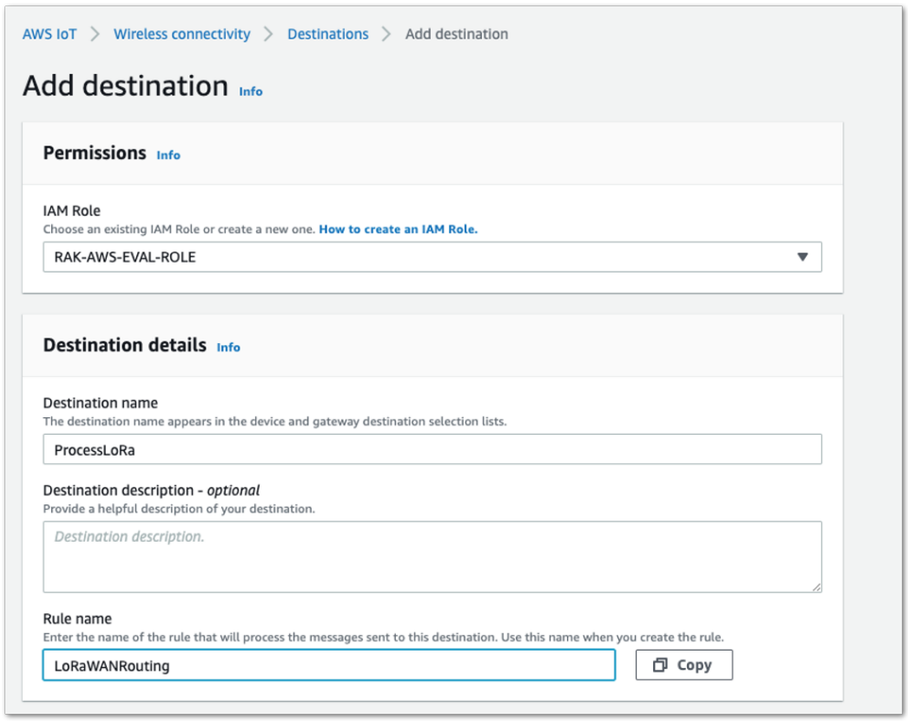

Figure 9: Adding Destination

4. Add Device.

Before adding a device to AWS IoT, retrieve the **DevEui**, **AppEui**, and **AppKey** from the end Device's console. You can use AT command `at+get_config=lora:status` to obtain the information.

For more AT commands, refer to the RAK4200 AT Command Manual.

```
at+get_config=lora:status\r\n
OK Work Mode: LoRaWAN
Region: EU868
Send_interval: 600s
Auto send status: false.
MulticastEnable: true.
Multi_Dev_Addr: 260111FD
Multi_Apps_Key: F13DDFA2619B10411F02F042E1C0F356
Multi_Nwks_Key: 1D1991F5377C675879C39B6908D437A6
Join_mode: OTAA
DevEui: 0000000000000888
AppEui: 0000000000000888
AppKey: 00000000000008880000000000000888
Class: C
Joined Network:false
IsConfirm: unconfirm
AdrEnable: true
EnableRepeaterSupport: false
RX2_CHANNEL_FREQUENCY: 869525000, RX2_CHANNEL_DR:0
RX_WINDOW_DURATION: 3000ms
RECEIVE_DELAY_1: 1000ms
RECEIVE_DELAY_2: 2000ms
JOIN_ACCEPT_DELAY_1: 5000ms
JOIN_ACCEPT_DELAY_2: 6000ms
Current Datarate: 4
Primeval Datarate: 4
ChannelsTxPower: 0
UpLinkCounter: 0
DownLinkCounter: 0
```

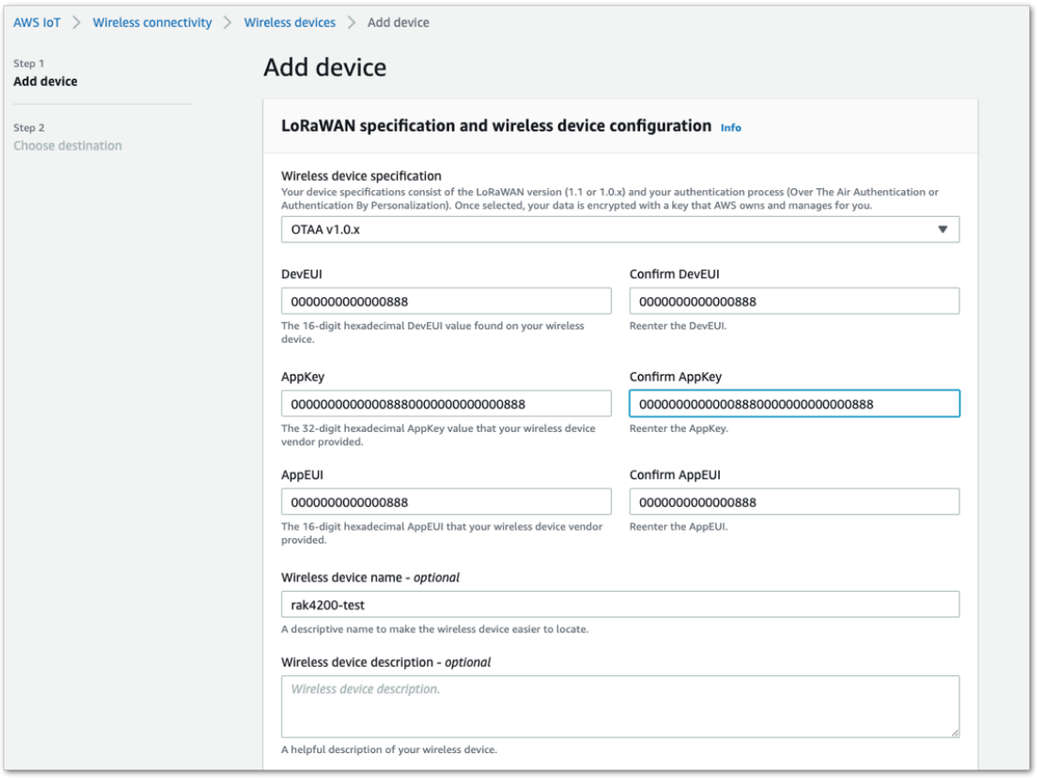

Figure 10: LoRaWAN specifications and wireless device configuration

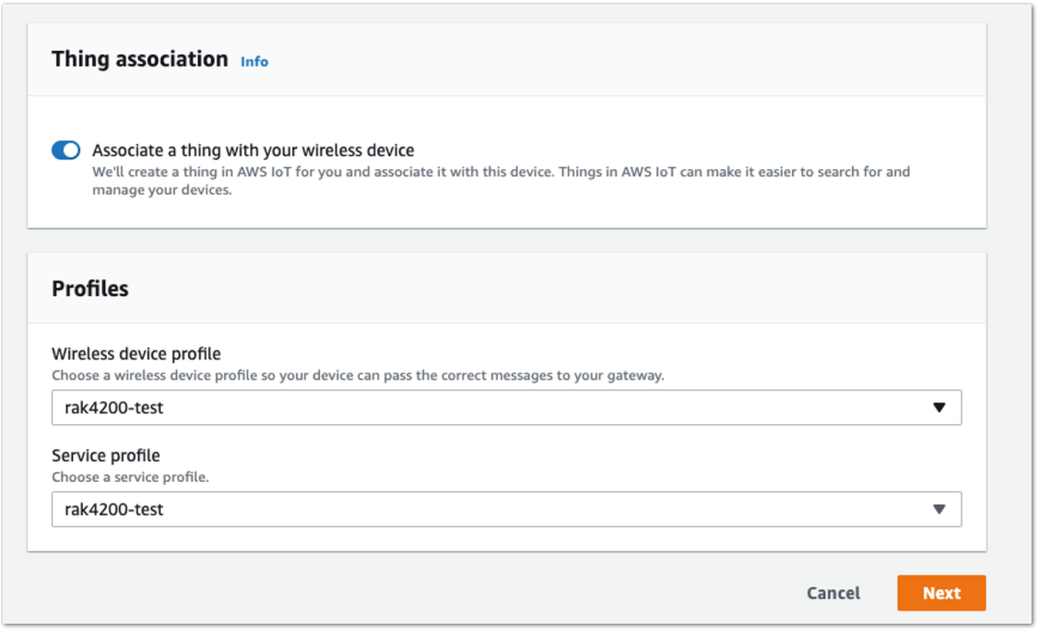

Figure 11: Choosing a Wireless Device Profile

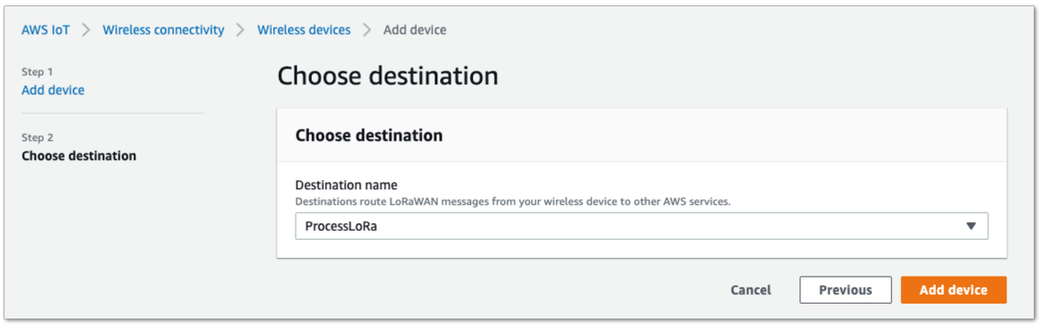

Figure 12: Choosing a Destination

5. Restart the end Device, and it should join the AWS IoT LoRaWAN server.

```
EVENT:0:STARTUP
SYSLOG:4:OTAA Join Request
SYSLOG:4:OTAA Join Success
EVENT:1:JOIN_NETWORK
SYSLOG:4:LoRa Tx :
```

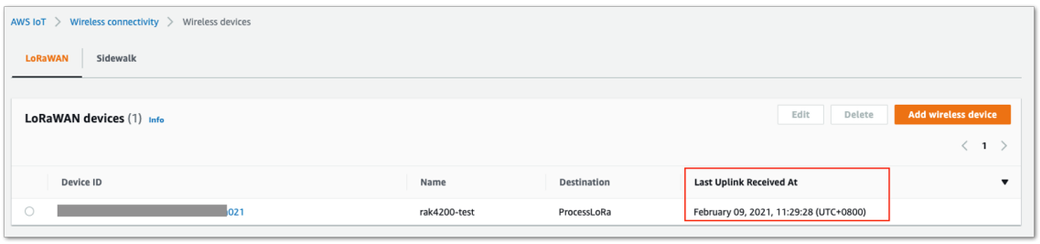

Figure 13: Choosing a Destination

6. Use the AT command `at+send:lora:1:1234567890` to send an uplink message.

Here is the console log after sending uplink message.

```
1
OK
SYSLOG:4:LoRa Tx : 1234567890
EVENT:3:LORA_TX_DONE:1:OK
```

Verifying Operation

Once setup is completed, provisioned OTAA devices can join the network and start to send messages. Messages from devices can then be received by AWS IoT Core for LoRaWAN and forwarded to the IoT Rules Engine.

Instructions for a sample Hello World application are given below, assuming that the device has joined and is capable of sending uplink traffic.

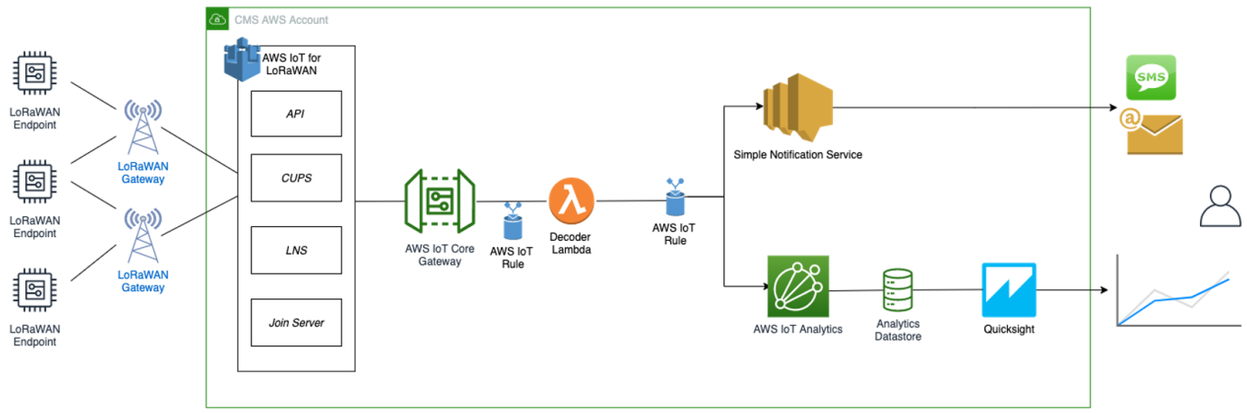

Figure 14: Sending Uplink Architecture

Create a Lambda Function for Destination Rule

Create the lambda function to process device messages processed by the destination rule.

1. Go to the AWS Lambda console .
2. Click on **Functions** in the navigation pane.
3. Click on **Create function**.
4. Select **Author** from scratch.
5. Under **Basic Information**, enter the function name and choose ***Runtime Python 3.8***. from the drop-down under **Runtime**.
6. Click on **Create function**.
7. Under **Function code**, paste the copied code into the editor under the ***lambda_function.py*** tab.

```py
import base64
import json
import logging
import ctypes
import boto3

# define function name

FUNCTION_NAME = 'RAK-HelloWorld'

# Second Byte in Payload represents Data Types
# Low Power Payload Reference: https://developers.mydevices.com/cayenne/docs/lora/

DATA_TYPES = 1

# Type Temperature

TYPE_TEMP = 0x67

# setup iot-data client for boto3

client = boto3.client('iot-data')

# setup logger

logger = logging.getLogger(FUNCTION_NAME)
logger.setLevel(logging.INFO)

def decode(event):
    data_base64 = event.get('PayloadData')
    data_decoded = base64.b64decode(data_base64)
    result = {
        'devEui': event.get('WirelessMetadata').get('LoRaWAN'
                ).get('DevEui'),
        'fPort': event.get('WirelessMetadata').get('LoRaWAN'
                ).get('FPort'),
        'freq': event.get('WirelessMetadata').get('LoRaWAN'
                ).get('Frequency'),
        'timestamp': event.get('WirelessMetadata').get('LoRaWAN'
                ).get('Timestamp'),
        }

    if data_decoded[DATA_TYPES] == TYPE_TEMP:
        temp = data_decoded[DATA_TYPES + 1] << 8 \
            | data_decoded[DATA_TYPES + 2]
        temp = ctypes.c_int16(temp).value
        result['temperature'] = temp / 10

    return result


def lambda_handler(event, context):
    data = decode(event)
    logger.info('Data: %s' % json.dumps(data))
    response = client.publish(topic=event.get('WirelessMetadata'
                              ).get('LoRaWAN').get('DevEui')
                              + '/project/sensor/decoded', qos=0,
                              payload=json.dumps(data))
    return response
```

8. Once the code has been pasted, choose **Deploy** to deploy the lambda code.
9. Click on the **Permissions** tab of the lambda function.
10. Change the **Lambda Role Policy** permission.
    - Under **Execution role**, click on the hyperlink under **Role name**.
    - On the **Permissions tab**, find the policy name and select it.
    - Choose **Edit policy**, and choose the **JSON** tab.
    - Append the following to the Statement section of the policy to allow publishing to AWS IoT.

```json
{
  "Effect":"Allow",
  "Action":[
     "iot:Publish"
  ],
  "Resource":[
     "*"
  ]
}
```

- Choose **Review Policy**, then Save changes.

11. Create a test event that will allow you to test the functionality of the lambda function.
    - In the drop-down, for the ***Select a test event***, choose **Configure test events**.
    - Enter a name for the test event under the **Event name**.
    - Paste the following sample payload in the area under Event name:

```json
    {
    "WirelessDeviceId": "65d128ab-90dd-4668-9556-fe47c589610b",
    "PayloadData": "AWf/1w==",
    "WirelessMetadata": {
    "LoRaWAN": {
    "DataRate": "4",
    "DevEui": "0000000000000088",
    "FPort": 1,
    "Frequency": "868100000",
    "Gateways": [
           {
    "GatewayEui": "80029cffXXXXXXXX",
    "Rssi": -109,
    "Snr": 5
           }
        ],
    "Timestamp": "2021-02-08T04:00:40Z"
        }
    }
    }
```

12. Choose **Create** to save the event.
13. Navigate to the AWS IoT console, choose **Test** on the navigation pane, and select **MQTT client**.
14. Configure the MQTT client to subscribe to "#" (all topics).
15. Click on **Test** in the Lambda function page to generate the test event you just created.
16. Verify the published data in the AWS IoT Core MQTT Test client:

    - Open another window. Go to **AWS IoT Console**, select **Test** under Subscription Topic, **enter #** and select to **Subscribe to topic**.

- The output should look similar to this:

```json
0000000000000000088/project/sensor/decoded      February 09, 2021, 14:45:29 (UTC+0800)
{
    "devEui": "00000000000000088",
    "fPort": 1,
    "freq": "868100000",
    "timestamp": "2021-02-08T04:00:40Z",
    "temperature": -4.1
}
```

Create the Destination Rule

In this section, create the IoT rule that forwards the device payload to your application. This rule is associated with the destination created earlier in Set up a Destination for Device Traffic section.

1. Navigate to the AWS IoT console .
2. In the navigation pane, choose **Act**, then select **Rules**.
3. On the Rules page, choose **Create**.
4. On the Create a rule page, enter as follows:
   - Name: **LoRaWANRouting**
   - Description: **Any description of your choice**.

> 📝 **NOTE:**
>
> The **Name of your Rule** is the information needed when you provision devices to run on AWS IoT Core for LoRaWAN.

5. Leave the default Rule query statement: '**SELECT * FROM 'iot/topic**' unchanged. This query has no effect at this time, as traffic is currently forwarded to the rules engine based on the destination.
6. Under Set one or more actions, choose **Add action**.
7. On the Select an action page, choose **Republish a message to an AWS IoT topic**. Scroll down and choose **Configure action**.
8. On the Configure action page, for Topic, enter ***project/sensor/decoded***.The AWS IoT Rules Engine will forward messages to this topic.
9. Under **Choose or create a role to grant AWS IoT access to perform this action**, select **Create Role**.
10. For Name, enter a name of your choice.
11. Choose **Create role** to complete the role creation. You will see a "**Policy Attached**" tag next to the role name, indicating that the Rules Engine has been permitted to execute the action.
12. Choose **Add action**.
13. Add one more action to invoke the Lambda function. Under **Set one or more actions**, choose **Add action**.
14. Choose **Send a message to a Lambda function**.
15. Choose **Configure action**.

16. Select the Lambda function created earlier and choose **Add action**.

17. Then, choose **Create rule**.

18. A "**Success**" message will be displayed at the top of the panel, and the destination has a rule bound to it.

You can now check that the decoded data is received and republished by AWS by triggering a condition or event on the device itself.

- Go to the AWS IoT console. In the navigation pane, select **Test**, and choose **MQTT client**.
- Subscribe to the wildcard topic '#" to receive messages from all topics.
- Send message from endDevice using AT command: `at+send:lora:1:01670110`.
- You should see traffic similar to that shown below.

```json
393331375d387505/project/sensor/decoded          February 09, 2021, 14:47:21 (UTC+0800)
{
"devEui": "393331375d387505",
"fPort": 1,
"freq": "867100000",
"timestamp": "2021-02-09T06:47:20Z",
"temperature": 27.2
}
```

```json
project/sensor/decoded    February 09, 2021, 14:47:21 (UTC+0800)
   {
       "WirelessDeviceID": "6477ec22-9570-31d5981da021",
       "PayloadData": "AWcBEA==",
       "WirelessMetadata": {
           "LoRaWAN": {
               "DataRate": "4",
               "DevEui": "393331375d387505",
               "FPort": 1,
               "Frequency": "867100000",
               "Gateways": [
                   {
                       "GatewayEui": "ac1ff09fffe014bd5",
                       "Rssi": -103,
                       "Snr": 8.5
                   }
               ],
               "Timestamp": "2021-02-09T06:47:20Z"
           }
       }
   }
```

Configuring Amazon SNS

You will be using the Amazon Simple Notification Service to send text messages (SMS) when certain conditions are met.

1. Go to the Amazon SNS console .
2. Click on the menu in the left corner to open the navigation pane.
3. Select **Text Messaging** (SMS) and choose **Publish text message**.
4. Under Message type, select **Promotional**.
5. Enter your phone number (phone number that will receive text alerts).

6. Enter "Test message" for the Message and choose **Publish** message.
7. If the phone number you entered is valid, you will receive a text message and your phone number will be confirmed.
8. Create an Amazon SNS Topic as follows:
   - In the navigation pane, choose Topics.
   - Select Create topic.
   - Under Details, select Standard.
   - Enter a name of your choice. Here, you will use "*text_topic*".
   - Choose Create topic.
9. Create a subscription for this topic:
   - On the page for the newly created text_topic, choose the **Subscriptions** tab.
   - Choose **Create subscription**.
   - Select **Protocol** as SMS from the drop-down.
   - Under Endpoint, enter the previously validated phone number to receive the SMS alerts.
   - Choose Create subscription. You should see a "***Subscription to text_topic created successfully***" message.

Add a Rule for Amazon SNS Notification

Now, add a new rule to send an Amazon SNS notification when certain conditions are met in a decoded message.

1. Navigate to the AWS IoT console .
2. In the navigation pane, choose **Act**. Then, choose **Rules**.
3. On the Rules page, choose **Create**.
4. Enter the Name as *text_alert* and provide an appropriate Description.
5. Under the **Rule query statement**, enter the following query:

```
SELECT devEui as device_id, "Temperature exceeded 25" as message, temperature as temp, timest
```

6. Choose **Add action**.
7. Choose **Send a message as an SNS push notification**.
8. Choose **Configure action**.
9. Under SNS target, select *text_topic* from the drop-down.
10. Select RAW under **Message format**.
11. Under **Choose or create a role to grant AWS IoT access to perform this action**, choose **Create role**.
12. Enter a name for the role and choose **Add action**.
13. Choose **Create rule**. You should see a "**Success**" message, indicating that the rule has been created.

Test the Rule for Amazon SNS Notification

After adding the rule for Amazon SNS notification, you should receive a text message when hitting the event.

Send message from endDevice using AT command: `at+send:lora:1:01670110`. Here is the message from mobile after sending an uplink message.

```json
{
    "device_id": "393331375d387505",
    "message": "Temperature exceeded 25",
    "temp": 27.2,
    "time": "2021-02-22T07:58:54Z"
}
```

Send Downlink Payload

This section shows how to send downlink payload from AWS IoT LoRaWAN Server to end Device.

1. Install the AWS SAM CLI .
2. Deploy SAM template to AWS .
3. Send Payload to End Device.
   - Go to the AWS IoT console.
   - In the navigation pane, select **Test**, and choose **MQTT client**.
   - Subscribe to the wildcard topic '#" to receive messages from all topics.
   - Specify the topic to ***cmd/downlink/{WirelessDeviceId}*** and a base64-encoded message.

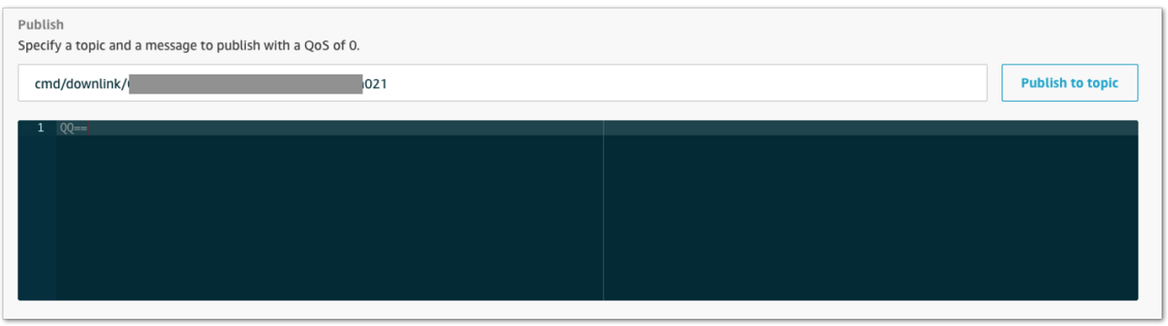

Figure 15: Specifying a topic

4. You should see traffic on AWS similar as shown below:

```
downlink/status/6477ec22-9570-4fea-9668-31d5981da021   February 09, 2021, 15:09:29 (UTC+08
{
    "sendresult": {
        "status": 200,
        "RequestId": "4f1d36e1-8316-4436-8e9d-2207e3711755",
        "MessageId": "60223529-0011d9f5-0095-0008",
        "ParameterTrace": {
            "PayloadDate": "QQ==",
            "WirelessDeviceId": "6477ec22-9570-4fea-9668-31d5981da021",
            "Fport": 1,
            "TransmitMode": 1
        }
    }
}
```

Figure 16: Traffic on AWS

5. You should see traffic on your console of end device similar as shown below.

```
SYSLOG:4:LoRa rX : 41 - 14
SYSLOG:4:LoRa Tx :
```

IoT Analytics

You will use IoT Analytics to visually display data via graphs if there is a need in the future to do further analysis.

Create an IoT Analytics Rule

Create a Rule First

1. Navigate to the AWS IoT console .
2. In the navigation pane, choose **Act** and then, choose **Rules**.
3. On the Rules page, choose **Create**.
4. Enter the Name as **Visualize**, and provide an appropriate Description.
5. Under the Rule query statement, enter the following query:

```
SELECT * FROM 'project/sensor/decoded'
```

6. Choose **Add action**.
7. Select **Send a message to IoT Analytics**.
8. Choose **Configure Action**.
9. Choose **Quick Create IoT Analytics Resources**.
10. Under **Resource Prefix**, enter an appropriate prefix for your resources, such as *LoRa Choose Quick Create*.
11. Once the Quick Create Finished message is displayed, choose **Add action**.
12. Choose **Create rule**. You should see a Success message, indicating that the rule has been created.

Configure AWS IoT Analytics

Set up AWS IoT Analytics

1. Go to the AWS IoT Analytics console .
2. In the navigation panel, choose **Data sets**.
3. Select the data set generated by the Quick Create in Create an IoT Analytics Rule
4. In the Details section, edit the **SQL query**.
5. Replace the query with as follows:

```
SELECT devEui as device_id, temperature as temp, timestamp as time FROM LoRa_datastore
```

6. Under Schedule, choose **Add schedule**.
7. Under Frequency, choose **Every 1 minute**, and then click **Save**.

Configure Amazon QuickSight

Amazon QuickSight lets you easily create and publish interactive BI dashboards that include Machine Learning-powered insights.

1. Go to AWS Management console .
2. From the management console, enter **QuickSight** in the "*Search for services, features..*" search box.

3. Click on QuickSight in the search results.
4. If you haven't signed up for the service before, go ahead and sign up, as there is a free trial period.
5. Select the **Standard Edition**, and choose Continue.
6. Enter a unique name in the field QuickSight account name.
7. Fill in the Notification email address.
8. Review the other checkbox options and change them as necessary. The **AWS IoT Analytics** option must be selected.
9. Choose **Finish**. You will see a confirmation message.
10. Choose **Go to Amazon QuickSight**.
11. Select **Datasets**.
12. Select **New dataset**.
13. Select **AWS IoT Analytics**.
14. Under Select an AWS IoT Analytics data set to import, choose the data set created in **Create an IoT Analytics Rule**.
15. Choose **Create data source**, and then choose **Visualize**.
16. Select the dataset created, then select **Refresh** or **Schedule Refresh** for a periodic refresh of the dataset.

Testing your "Hello Word" Application

Using your device, create a condition to generate an event such as a high-temperature condition. If the temperature is above the configured threshold then you will receive a text alert on your phone. This alert will include key parameters about the alert.

You can also visualize the data set as follows:

1. Go to the AWS IoT Analytics console .
2. Choose **Data sets**.
3. Select the dataset created earlier.
4. Select **Content** and ensure there are at least few uplink entries available in the data set.
5. Go to the **QuickSight console** .
6. Choose **New analysis**.
7. Choose the dataset created in **Create an IoT Analytics Rule**.
8. Select time on the X-axis, Value as temp (Average) and Color as device_id to see a chart of your dataset.

Debugging

After login to the device using the web browser, the system log can be viewed from **Status** > **System Log**.

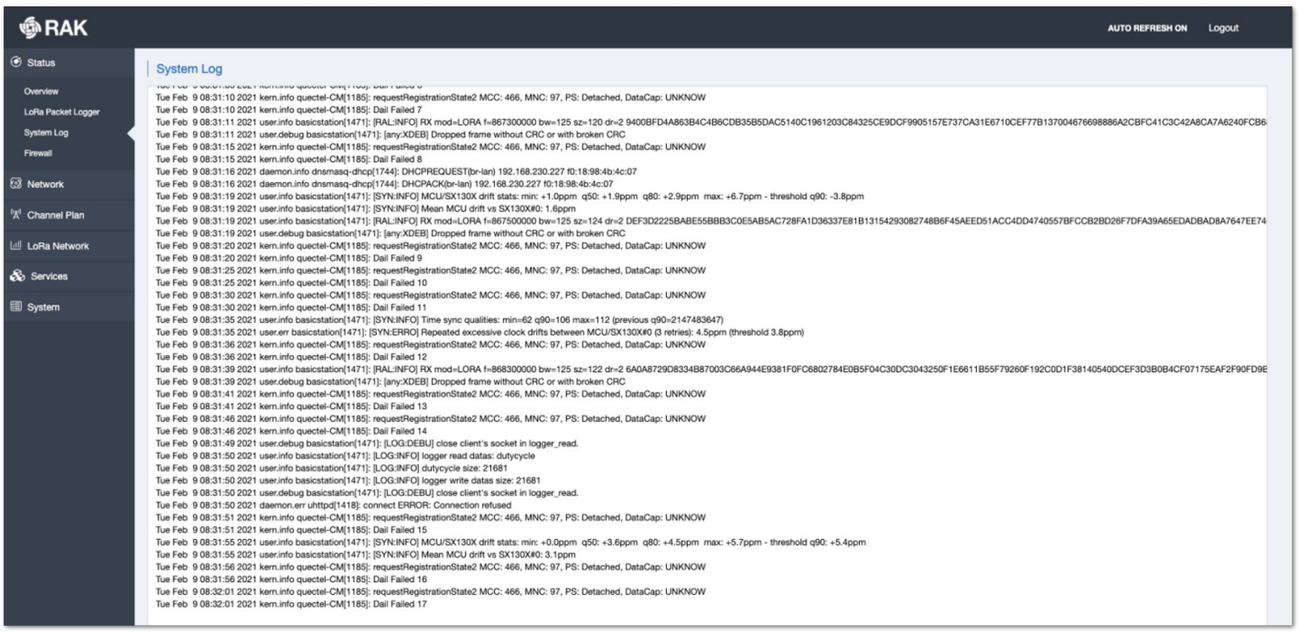

Figure 17: System Log

Troubleshooting

1. Unable to see the web login:
   - Check that your wifi is connected to **RAK7249_XXXX**.
   - Try ping **192.168.230.1**.
2. Lost password to login to the web login.
   - Hold the reset button for 10 seconds to factory reset the device

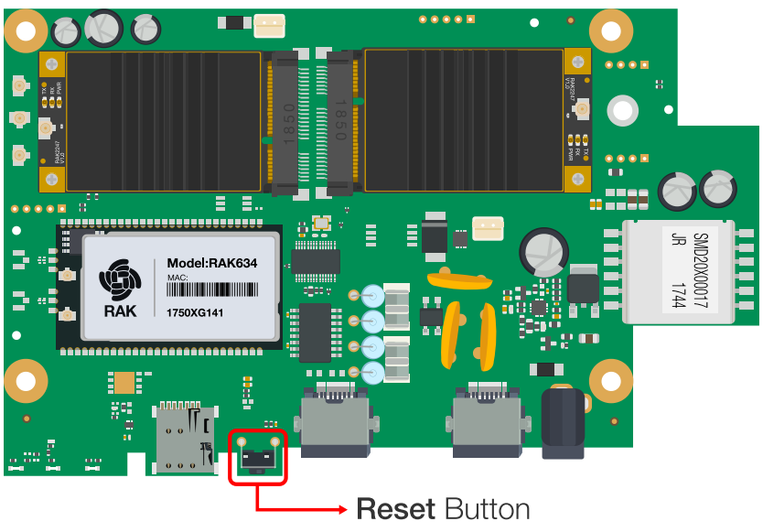

Figure 18: Troubleshooting

The Things Network (TTN)

Connecting to TTNv3

At The Things Conference 2021, it was announced that The Things Network is upgrading to The Things Stack v3.

In this section, it will be shown how to connect RAK7249 WisGate Edge Max to TTNv3.

To login into the TTNv3, head on to the TTN site. If you already have a TTN account, you can use your The Things ID credentials to log in.

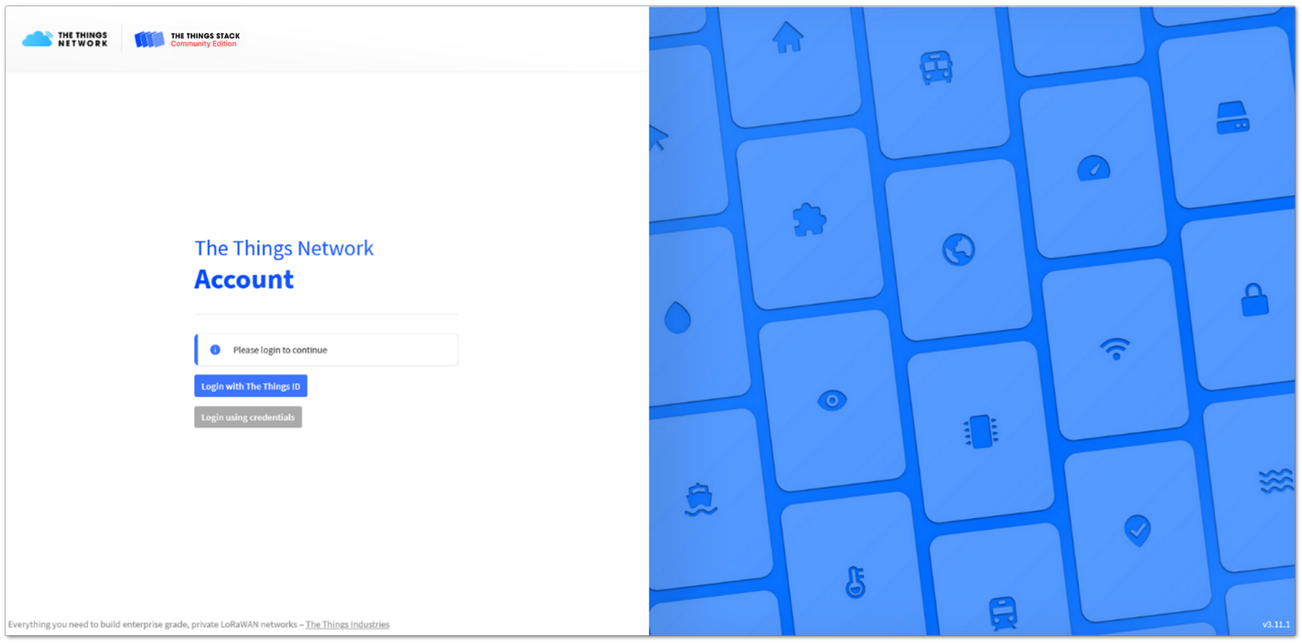

Figure 19: The Things Stack Home Page

> 📝 **NOTE:**
>
> This tutorial is for the EU868 Frequency band.

Register the Gateway

1. To register a commercial gateway, choose **Register a gateway** (for new users that do not already have a registered gateway) or go to **Gateways** > **+ Add gateway** (for users that have registered gateways before).

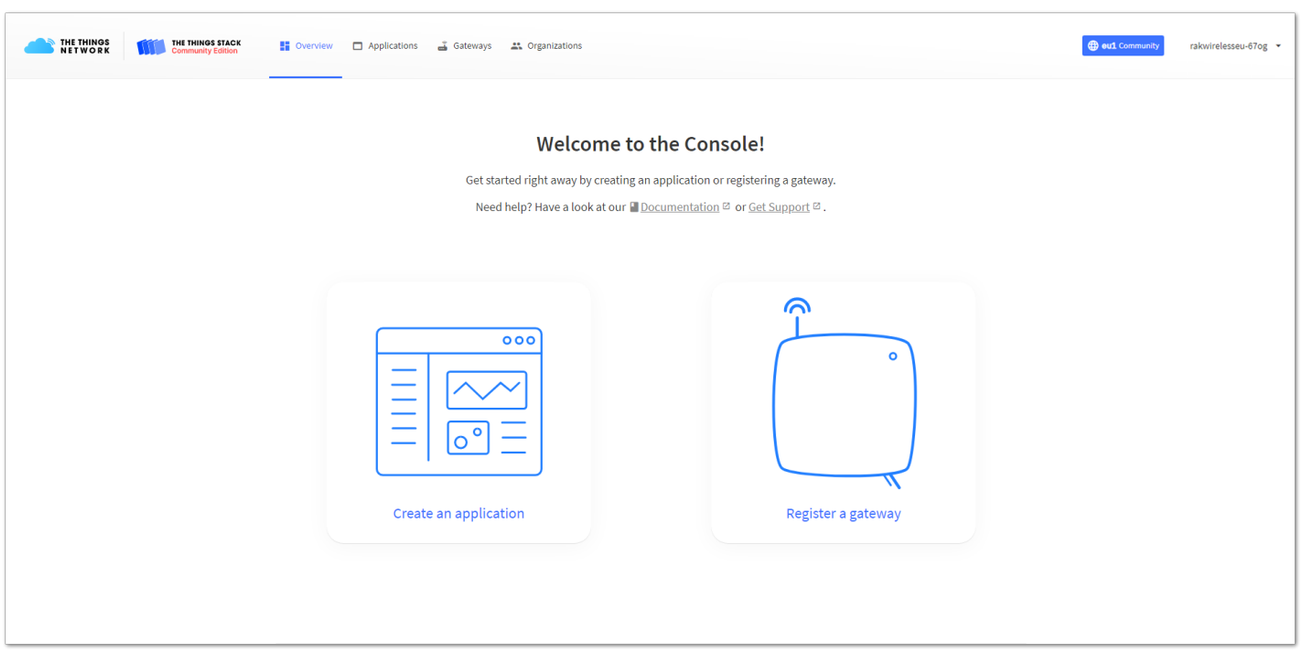

Figure 20: Console Page afer a successful login

2. Fill in the needed information:

- **Owner**: Automatically filled by The Things Stack, based on your account or created Organization.
- **Gateway ID**: This will be the unique ID of your gateway in the Network. Note that the ID must contain only lowercase letters, numbers, and dashes (-).
- **Gateway EUI**: A 64 bit extended unique identifier for your gateway. The gateway's EUI can be found either on the sticker on the casing or by going to the **LoRa Network Settings** page in the **LoRa Gateway** menu accessible via the Web UI. Instructions on how to access your gateway via Web UI can be found in the Quick Start Guide .
- **Gateway name**: A name for your gateway.
- **Gateway description (optional)**: Optional gateway description; can also be used to save notes about the gateway.
- **Gateway Server address**: The address of the Gateway Server to connect to.

> 📝 **NOTE:**
>
> This tutorial is based on using the EU868 frequency band, so the server address will be: **eu1.cloud.thethings.network**.

- **Frequency plan**: The frequency plan used by the gateway.

> 📝 **NOTE:**
>
> For this tutorial, the Europe 863-870 MHz (SF9 for RX2 - recommended) is used.

- The other settings are optional and can be changed to satisfy your requirements.

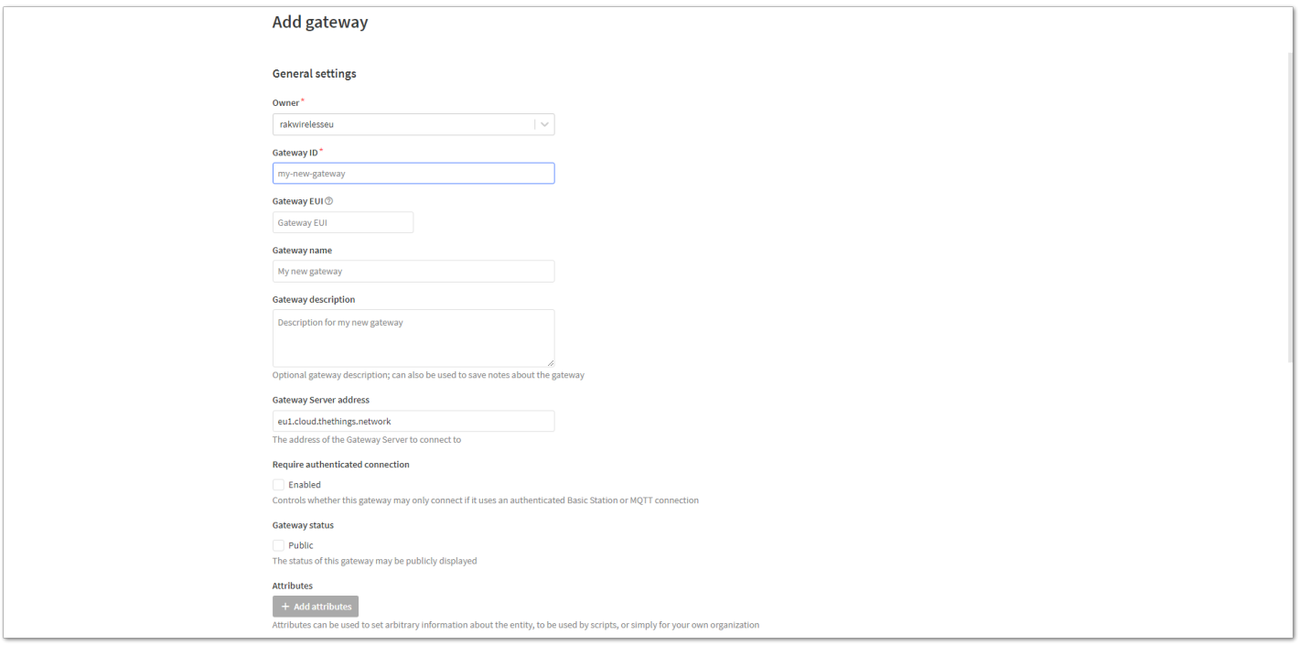

Figure 21: Adding a gateway

3. To register your gateway, scroll down and click **Create gateway**.

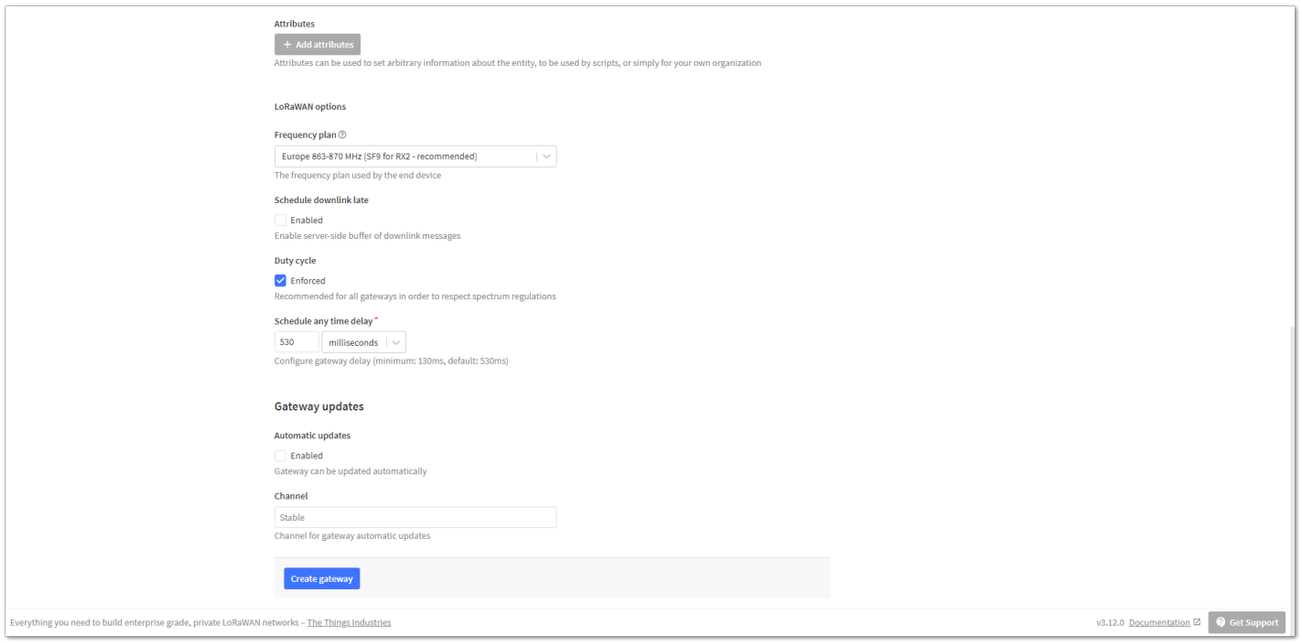

Figure 22: Registering the gateway

TTNv3 supports TLS server authentication and Client token, which requires a trust file and a key file to configure the gateway to successfully connect it to the network.

Generating the Token

1. To generate a key file, from the **Overview page** of the registered Gateway navigate to **API keys**.

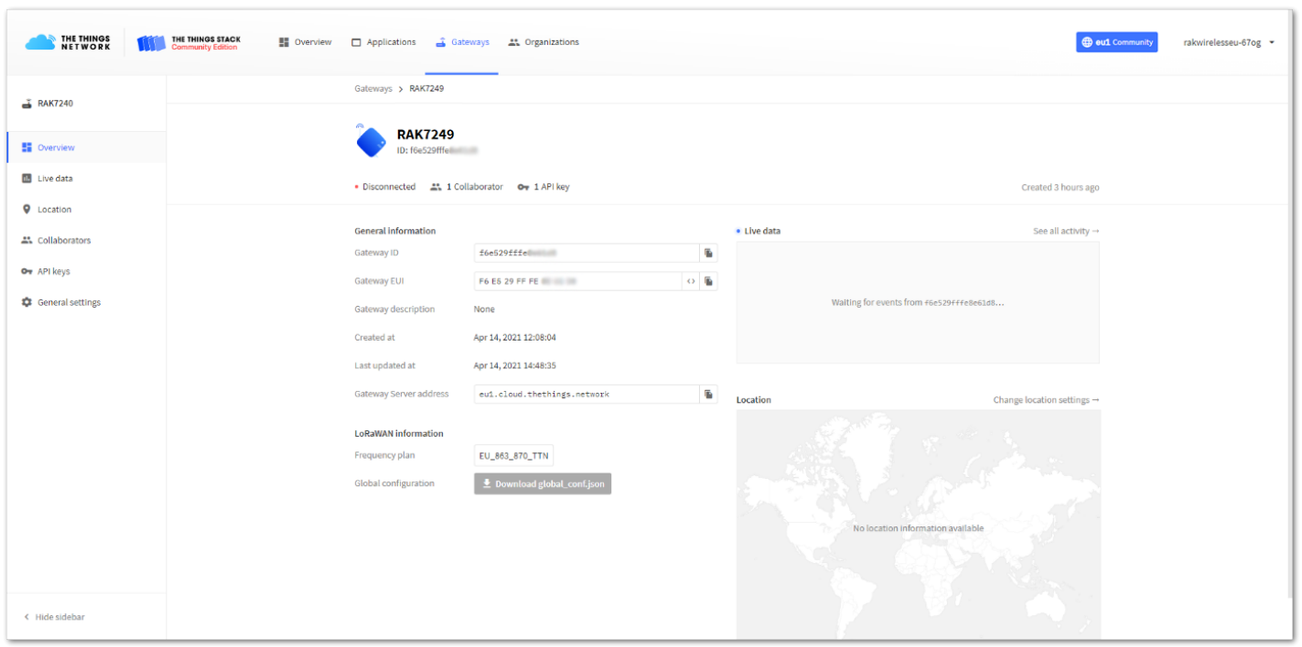

Figure 23: Overview page

2. In the **API keys page**, choose **+ Add API key**.

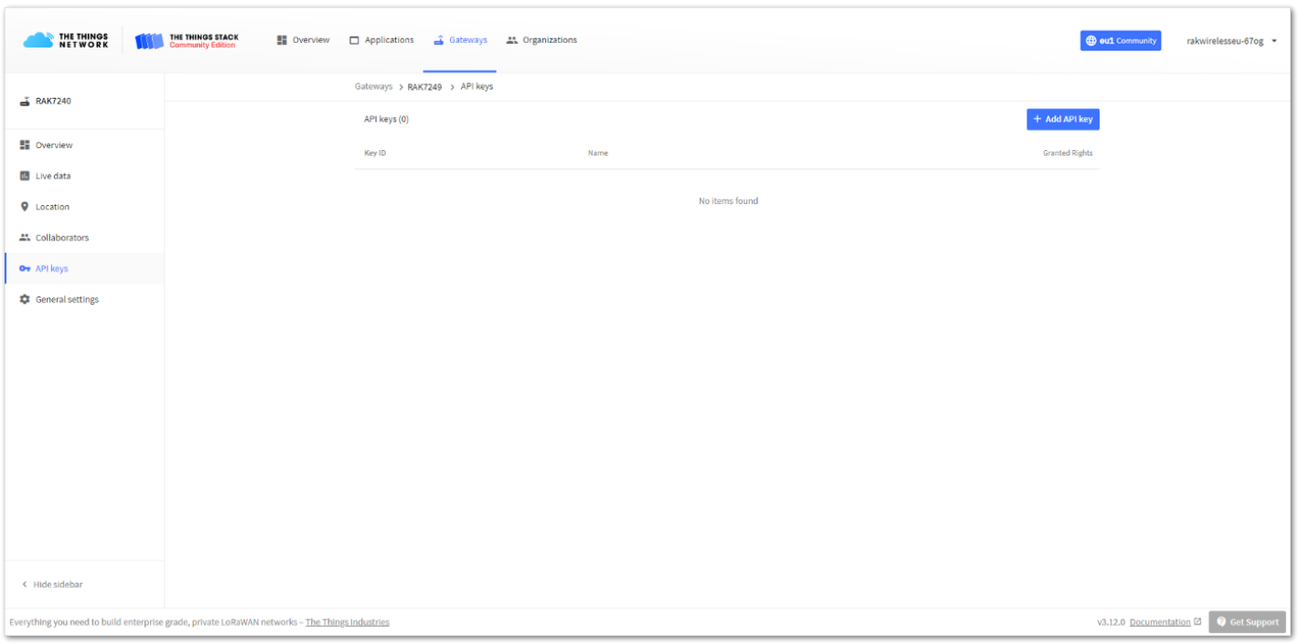

Figure 24: API key page

3. In the **Name field**, type the name of your key (for example - mykey). Choose **Grant individual rights** and select **Link as Gateway to a Gateway for traffic exchange, i.e. read uplink and write downlink**.

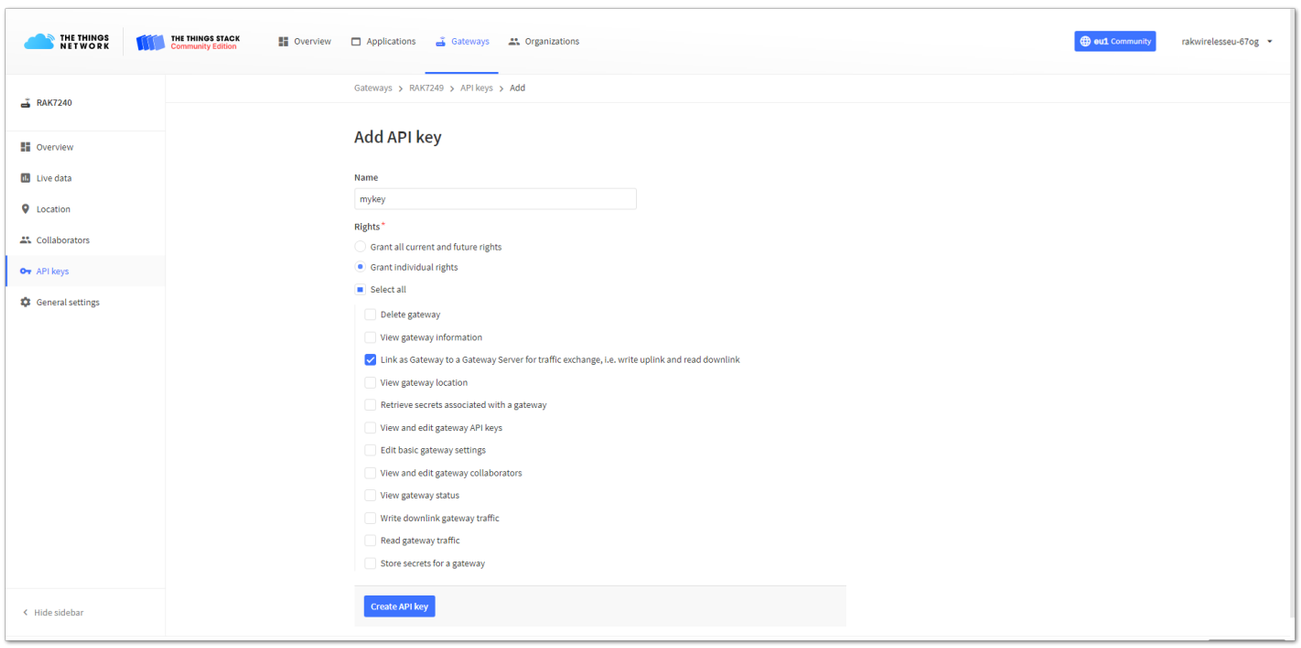

Figure 25: Generating an API key

4. To generate the key, choose **Create API key**. The following window will pop up, telling you to copy the key you just generated.

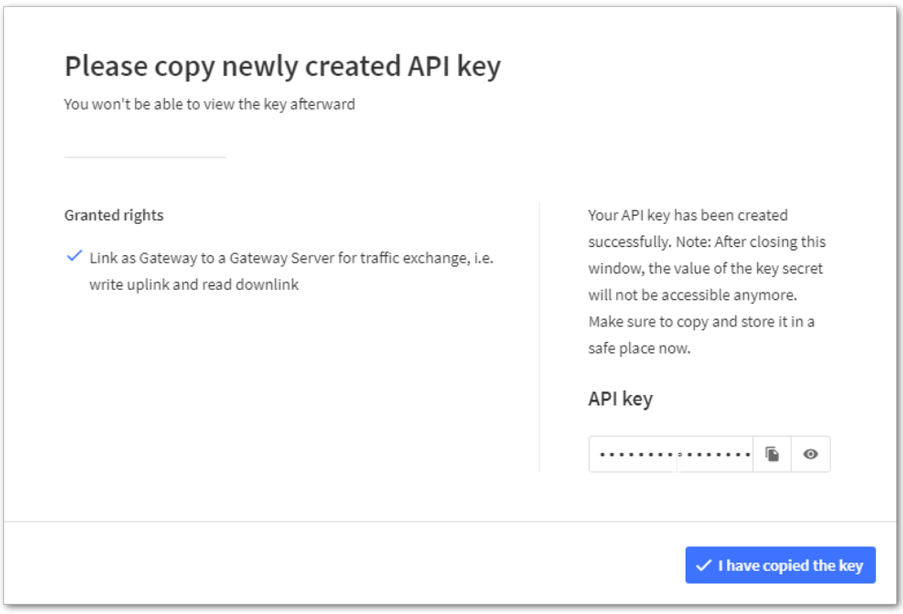

Figure 26: Copying the generated key

> ⚠ **WARNING**
>
> Copy the key and save it in a .txt file (or other), because you won't be able to view or copy your key after that.

5. Click **I have copied the key** to proceed.

Configuring the Gateway

1. To configure the gateway access it via the Web UI. To learn how to do that, refer to the Quickstart Guide .

2. Navigate to **LoRa Network** > **Network Settings** > **Mode** drop-down menu > choose **Basics Station**.

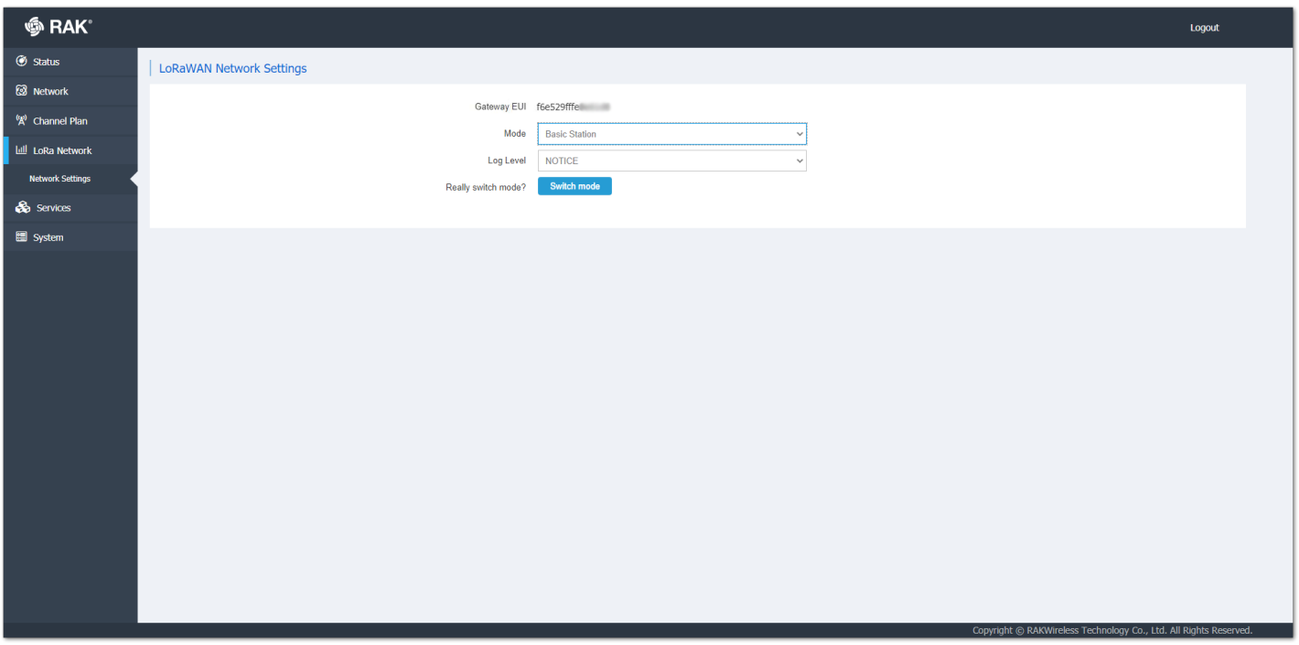

Figure 27: Changing the working mode

3. Select **Switch mode** to apply the change. After that, the **Basics Station Configuration** pane settings will show up. To connect the Gateway to TTNv3, the following parameters must be configured:

- **Server**: For server, choose **LNS Server**.
- **URI**: This is the link to The Things Stack server.

> 📝 **NOTE:**
>
> For this tutorial, you are connecting the gateway to the European cluster. For Europe, fill in the following: **wss://eu1.cloud.thethings.network**.

- **Port**: The LNS Server uses port 8887. Type in **8887**.
- **Authentication Mode**: Choose **TLS server authentication and Client token**. When selected, the trust and the token field will show up.
- **trust**: For trust, use the **Let's Encrypt ISRG ROOT X1 Trust** certificate. You can download and check the certificate .
- **token**: This is the generated **API key**. The key must start with **Authorization:**.

For example,

```
Authorization: YOUR_API_KEY
```

> 📝 **NOTE:**
>
> Replace **YOUR_API_KEY** with the key generated previously. Have in mind that there should be a "**space**" between **Authorization:** and **YOUR_API_KEY**, as shown in the example.

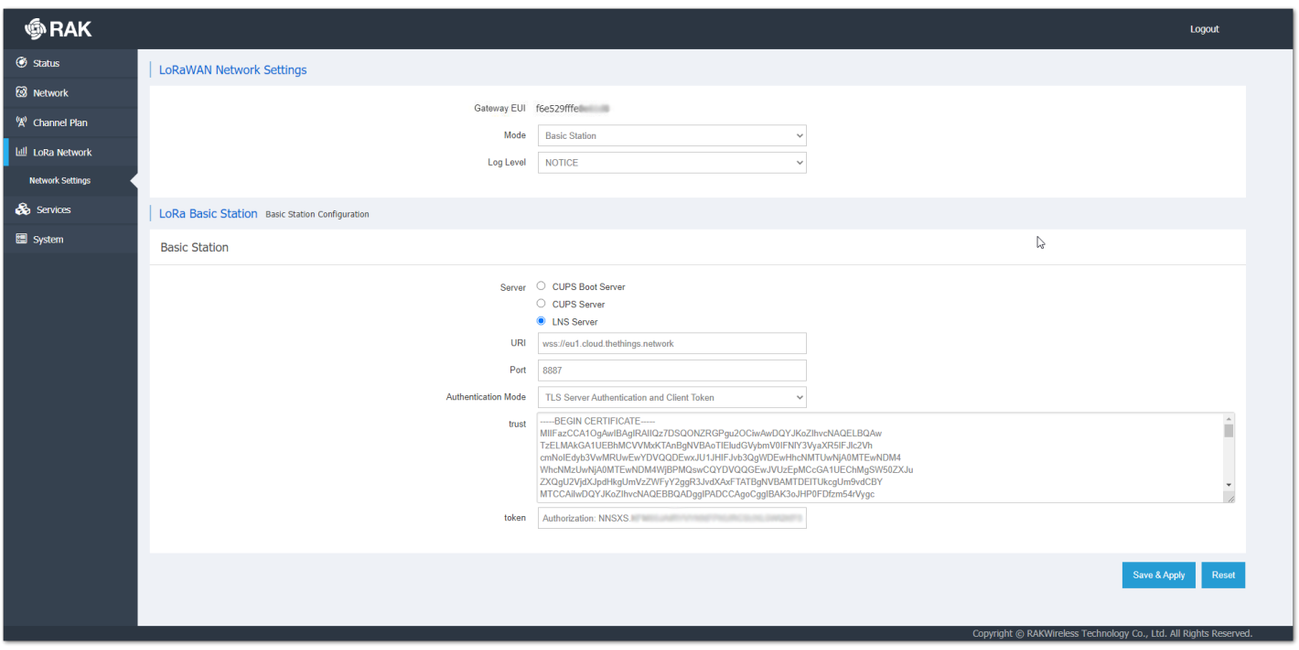

Figure 28: LoRa Basics Station settings

4. To save the changes, click **Save & Apply**.

You can now see that your gateway is connected to TTNv3 as Basics Station.

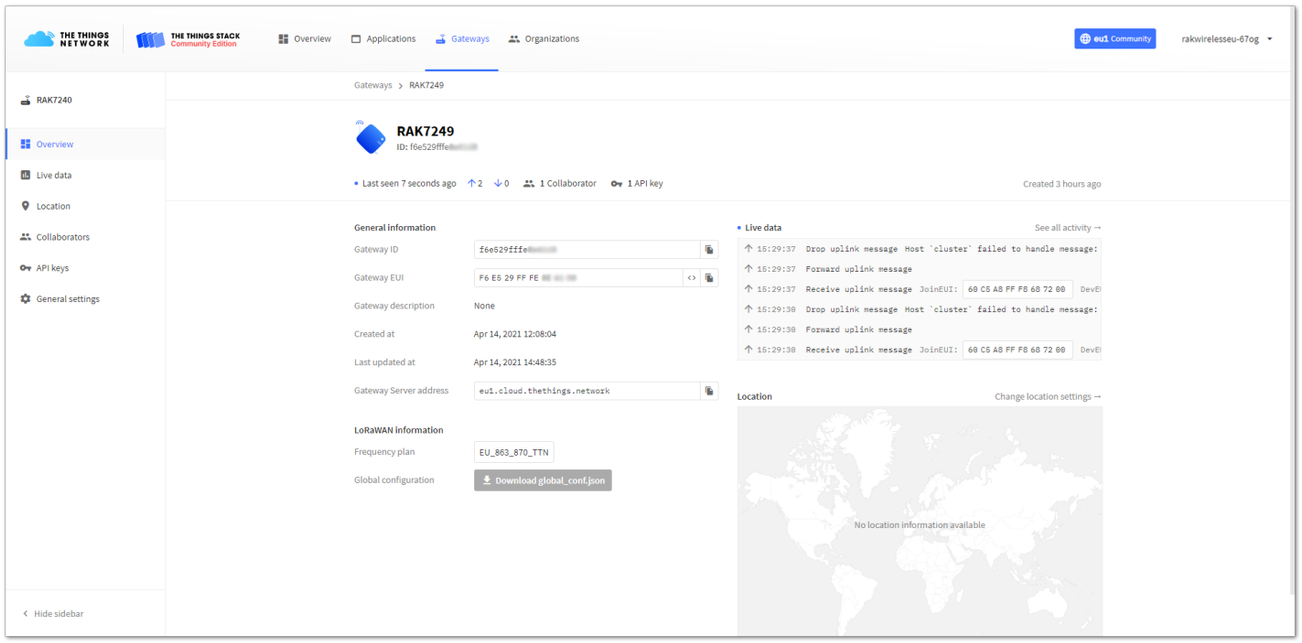

Figure 29: Successful connection

LORIOT

In this tutorial, you will learn how to connect RAK7249 WisGate Edge Max to LORIOT.

LORIOT provides an easy-to-use software platform that enables you to build, operate, and scale a secure IoT network suitable for long-range IoT solution deployments in every part of the world.

Prerequisites

Hardware

- RAK7249 WisGate Edge Max

Software

- SSH Client (This tutorial will be done using PuTTY .)
- LORIOT Account

Registering the Gateway

1. Log into your LORIOT account.

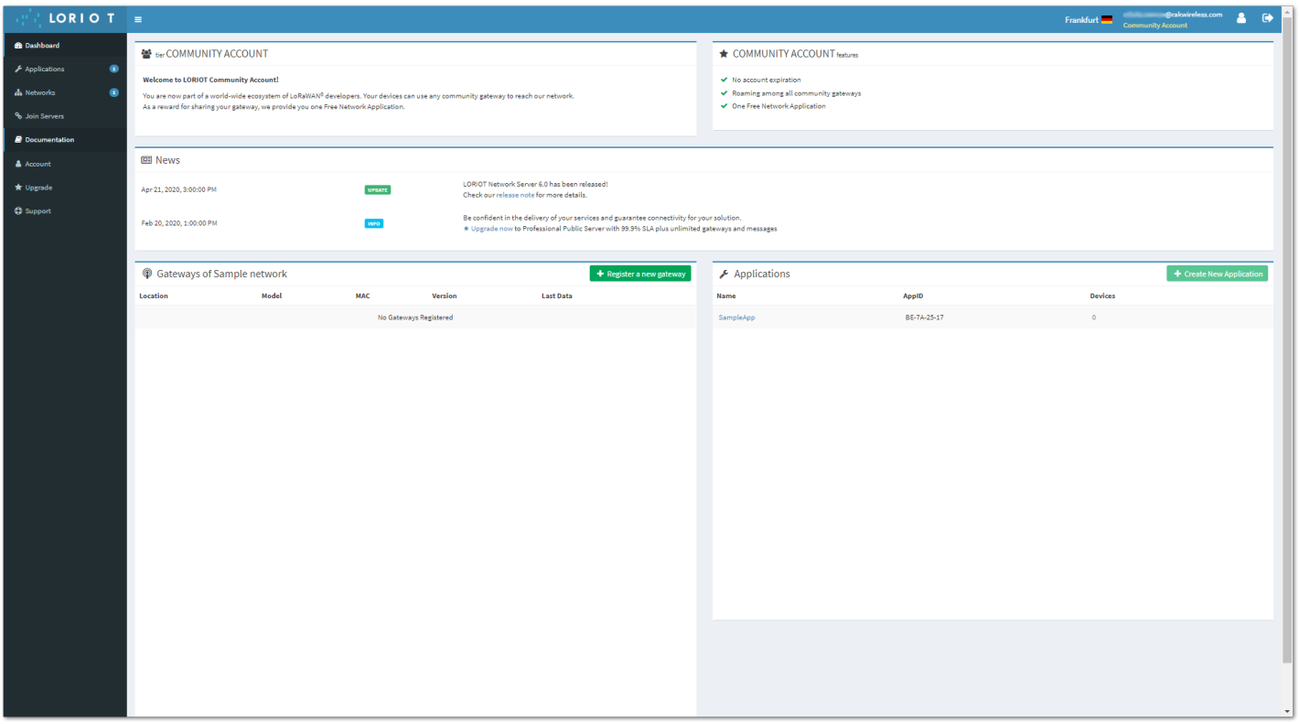

Figure 30: LORIOT Homepage

2. Go to the **Networks** tab of the main menu on the left. You have the option to select **Simple network**, which is automatically generated when you create your account, or you can create a new one to use. For a beginner, it will be easier to use the **Simple network**.

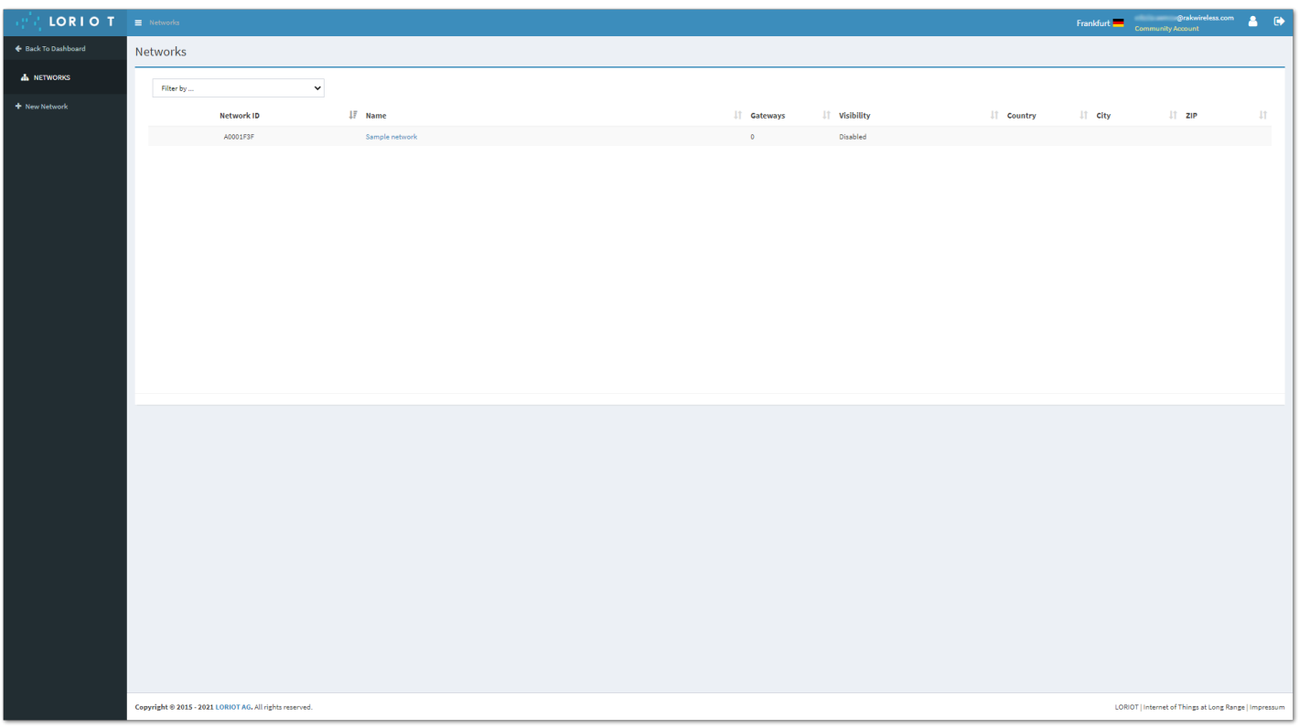

Figure 31: Networks List

3. Open the network by clicking once on its name. Then, click the **+ Add Gateway** button.

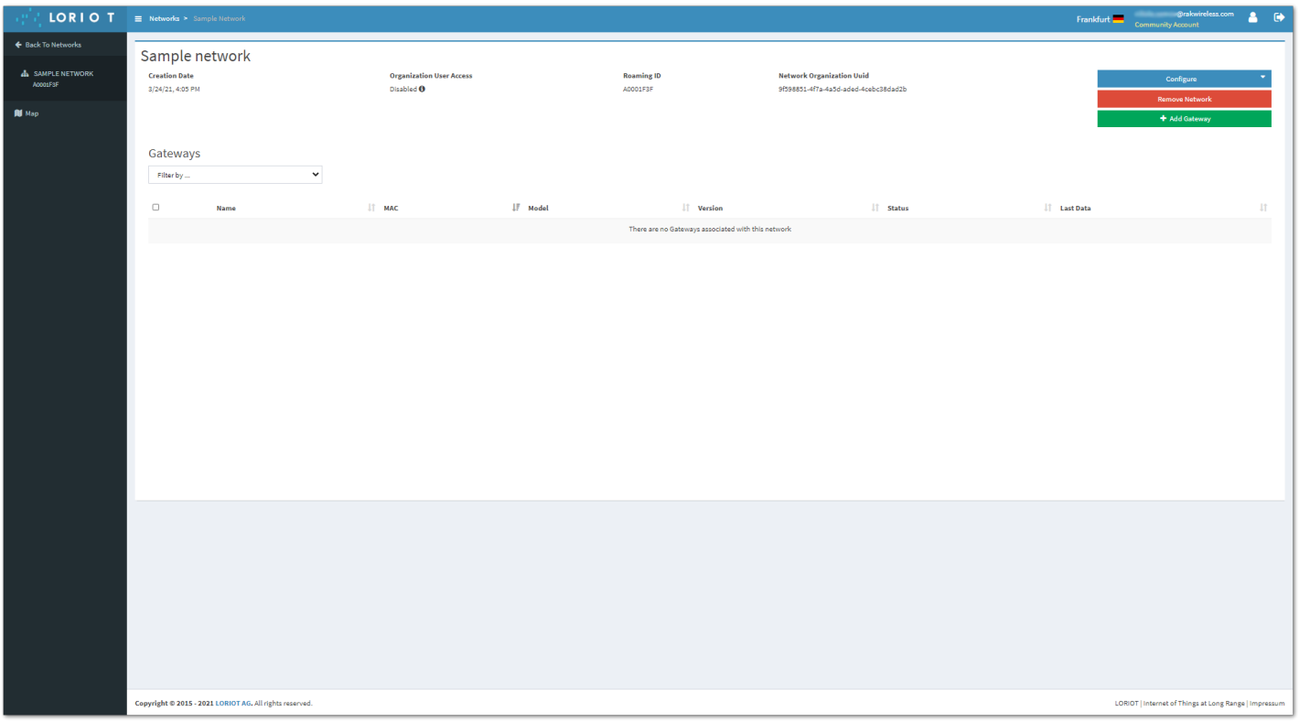

Figure 32: Adding a gateway to the network

4. In the list of gateways, find and select RAK7249.

> **NOTE**
>
> If you are using another model gateway from the WisGate Edge series, you still need to select RAK7249 in this list. This won't affect the performance in any way.

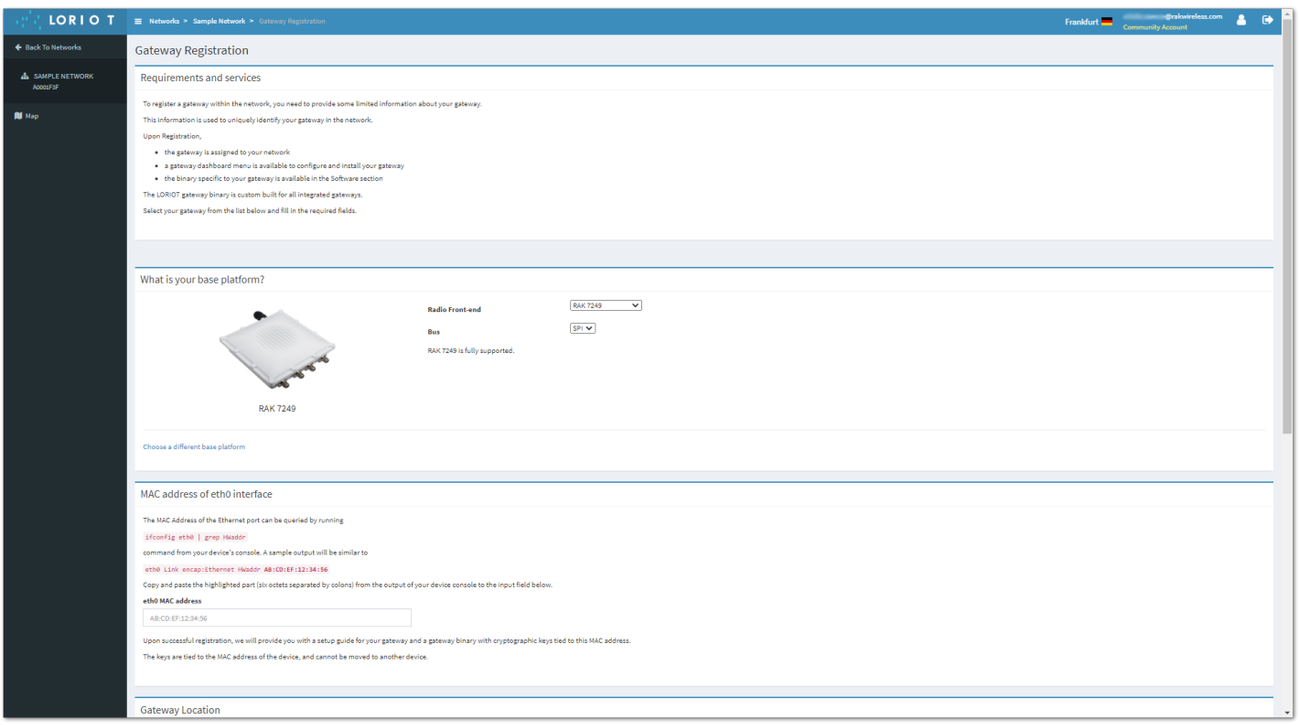

Figure 33: Selecting RAK7249

5. Now, you need to connect to your gateway via SSH. As mentioned, this tutorial will be done with the PuTTY SSH client. Open PuTTY and enter the IP address of your gateway. If your gateway is in AP mode, the address will be **192.168.230.1**.

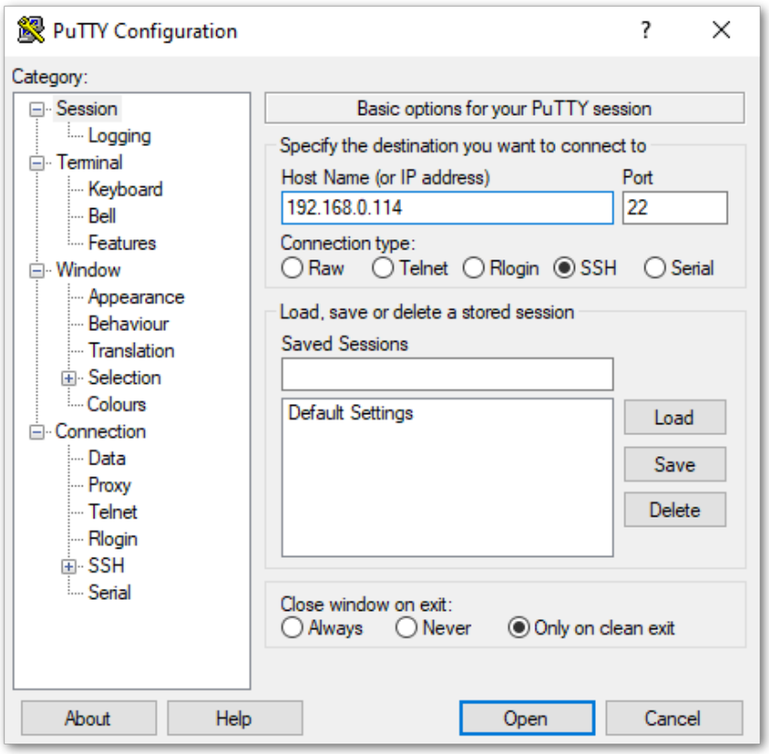

Figure 34: PuTTY Configuration

6. Log in with your root credentials.

- Default username: **root**
- Password: **root**

To get the MAC address of your gateway, run the command:

```
ifconfig eth0 | grep HWaddr
```

The output should be similar to the following:

```
eth0      Link encap:Ethernet  HWaddr 60:C5:A8:XX:XX:XX
```

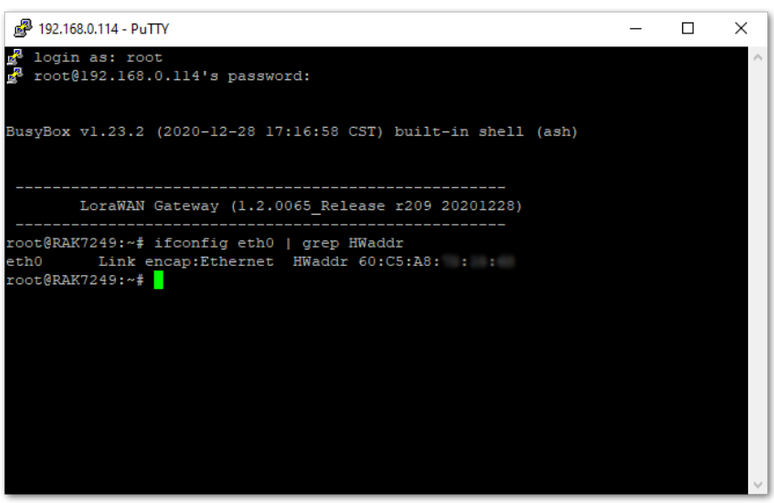

Figure 35: Getting the MAC address of the gateway

7. Copy the MAC address and fill it out in the registration form for the gateway in LORIOT. Scroll down and press the **Register RAK7249 gateway** button.

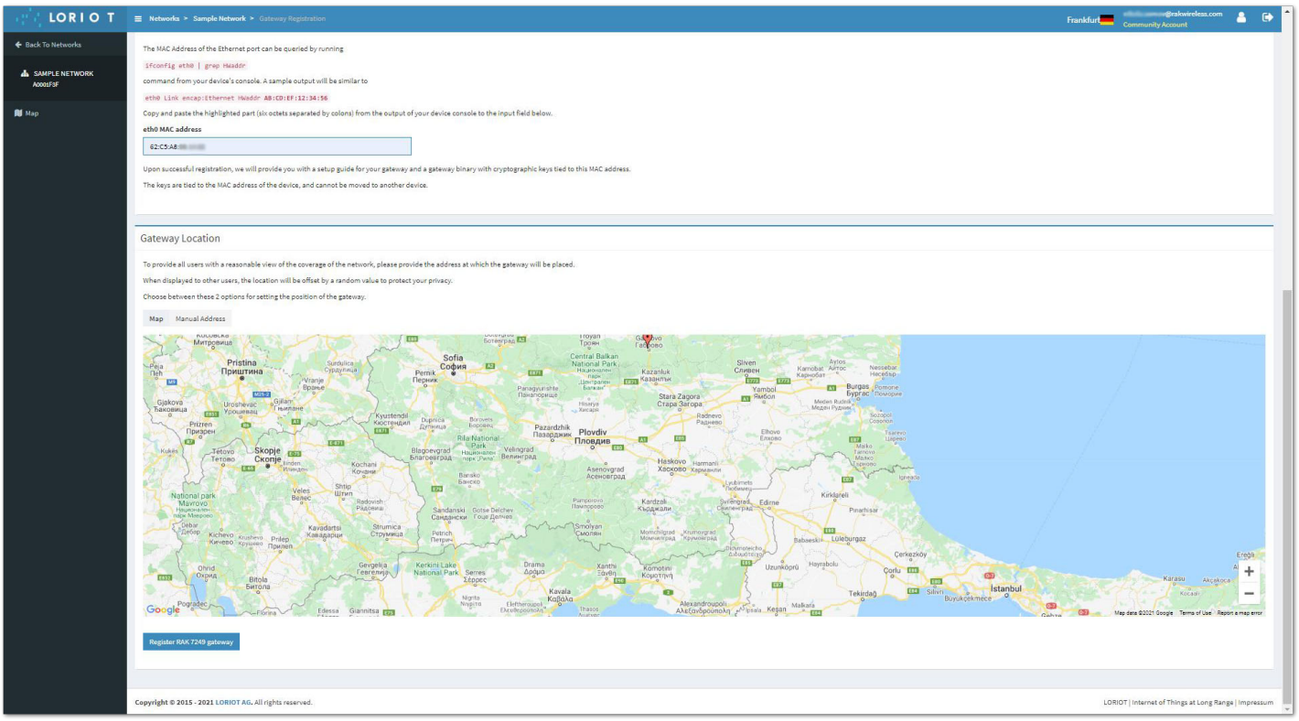

Figure 36: Filling out the MAC address

8. The gateway is now registered and you need to add a security layer to the connection. It is provided by LORIOT's Gateway Software. To get it installed, run the following set of commands in the PuTTY.

```
cd /tmp
```

```
wget http://eu1.loriot.io/home/gwsw/loriot-rak-7249-SPI-0-latest.sh -O loriot-install.sh
```

```
chmod +x loriot-install.sh
```

```
./loriot-install.sh -f -s eu1.loriot.io
```

```
/etc/init.d/packet_forwarder disable ; /etc/init.d/loriot-gw enable ; reboot now
```

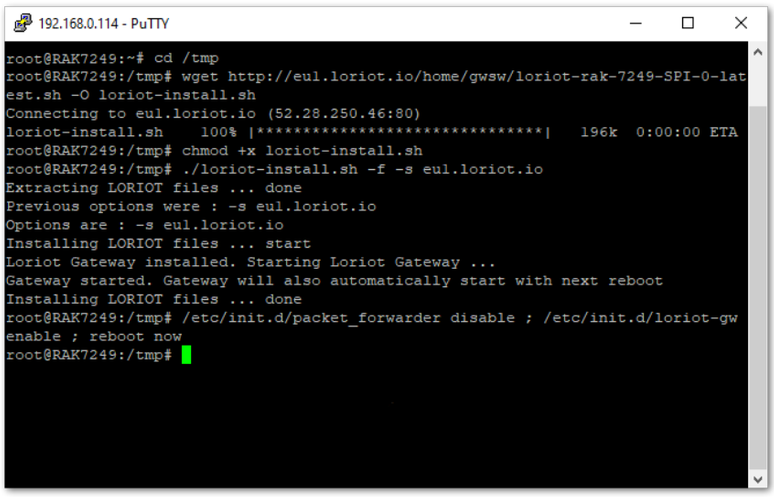

Figure 37: Installing LORIOT software

Your gateway is now registered and connected to LORIOT.

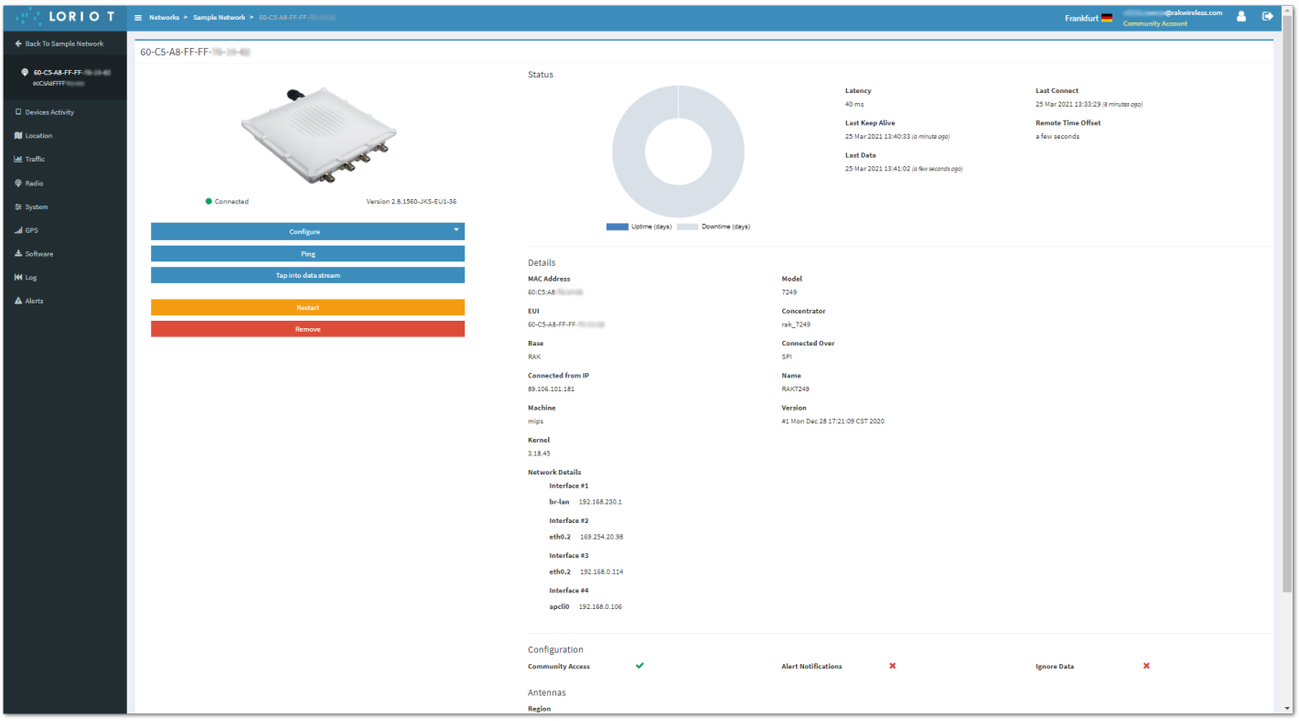

Figure 38: Successful Connection

Last Updated: 9/17/2021, 11:16:42 AM