RAK7268 Supported LoRa Network Servers

AWS IoT Core for LoRaWAN

If you don't have an AWS account, refer to the instructions in the guide here . The relevant sections are Sign up for an AWS account and Create a user and grant permissions.

Overview

The high-level steps to get started with AWS IoT Core for LoRaWAN are as follows:

1. Onboard your Gateway (see section Add the Gateway to AWS IoT)
2. Onboard your Device(s) (see section Add a LoRaWAN Device to AWS IoT) a. Verify device and service profiles b. Set up a Destination to which device traffic will be routed and processed by a rule.

These steps are detailed below. For additional details, refer to the AWS LoRaWAN developer guide .

Add the Gateway to AWS IoT

Preparation

Refer to the online guide for steps required prior to onboarding your gateway. For more details check the datasheet page: WisGate Edge Lite 2 Datasheet Software .

Frequency Band selection and Role setup

Refer to the online guide for information on selecting an appropriate frequency band.

> 📝 **NOTE**
>
> LoRa® Frequency bands supported by RAK7248: IN865, EU868, US915, AU915, KR920 and AS923 , please select an appropriate frequency band from our Store

Follow the instructions in the section Add an IAM role to allow the Configuration and Update Server (CUPS) to manage gateway credentials in the online guide .

Add the LoRaWAN Gateway

To register the Gateway with AWS IoT Core for LoRaWAN, follow the steps in this online guide under the section Add a gateway using the console.

Add a LoRaWAN Device to AWS IoT

Preparation

Refer to the datasheet to learn more about RAK4631 WisBlock LPWAN Module . Refer to the instructions in the section Before onboarding your wireless device in the online guide . Then follow the instructions in the section Add your wireless device to AWS IoT Core for LoRaWAN here .

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. For more details, refer to the section Add profiles to AWS IoT Core for LoRaWAN in the online guide .

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, refer to the online guide (sections titled Add a destination using the console and Create an IAM role for your destinations). Also refer to Create rules to process LoRaWAN device messages in the online guide .

Set up the Gateway

- Set up Gateway Hardware: Refer to the product configuration to learn more about RAK7268 Product Configuration .
- Set up Gateway Software: Refer to the product configuration to learn more about RAK7268 Product Configuration .
- Additional Software References:
  - FAQ
  - Forum

Configuring the Gateway device with WisGateOS 1

Figure 1: RAK7268 With WisGate OS 1 LoRaGateway Setting

By default, the Gateway will work in Wi-Fi AP Mode which means that you can find an SSID named like "RAK7268_XXXX" on your PC's Wi-Fi Network List. "XXXX" is the last two bytes of the Gateway MAC address. To access the Web Management Platform, input the IP Address: 192.168.230.1 in your Web browser.

(Note: No password is required to connect via Wi-Fi.)

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

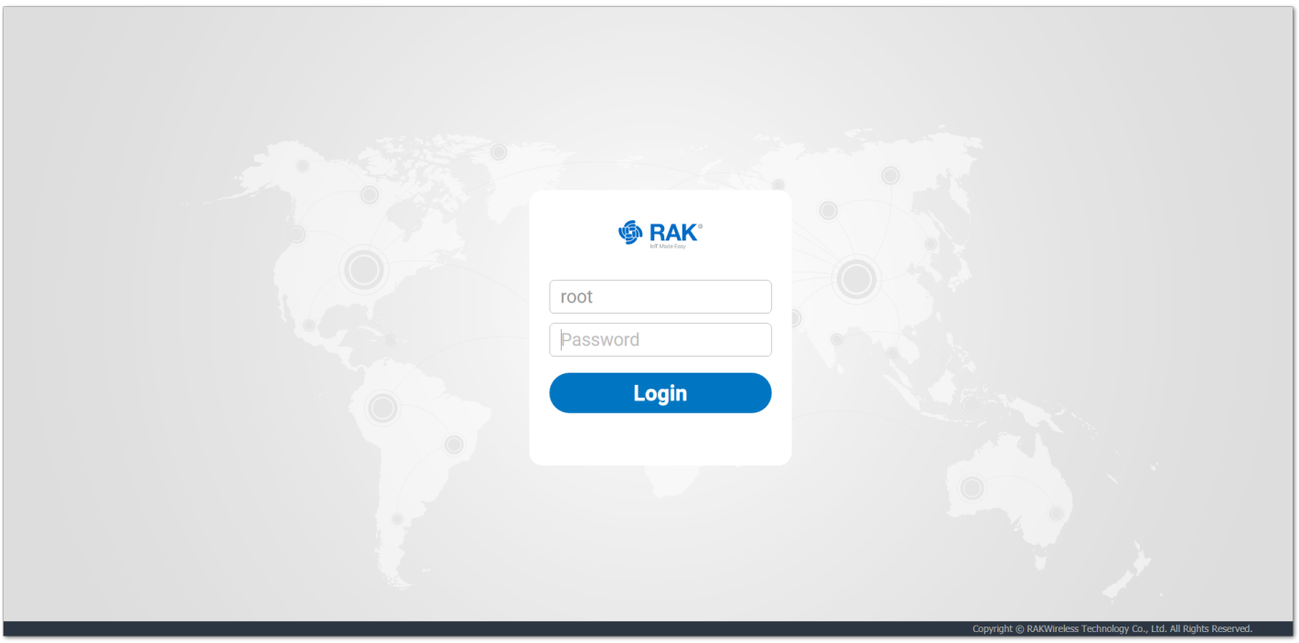

Figure 2: Web User Interface Log-in

C:\Users\Mark\Documents\Work\RAKwireless\Documentation\rakwireless-docs-internal\docs.vuepress\public\assets\images\wisgate\rak7268\supported-lora-network-servers\aws The first firmware version that supports AWS IoT Core for LoRaWAN is 1.2.0065_Release_r209, it can be verified on Status -> Overview -> System -> Firmware Version.

Navigate to System -> Backup/Flash Firmware -> Flash new firmware image, and upgrade the firmware.

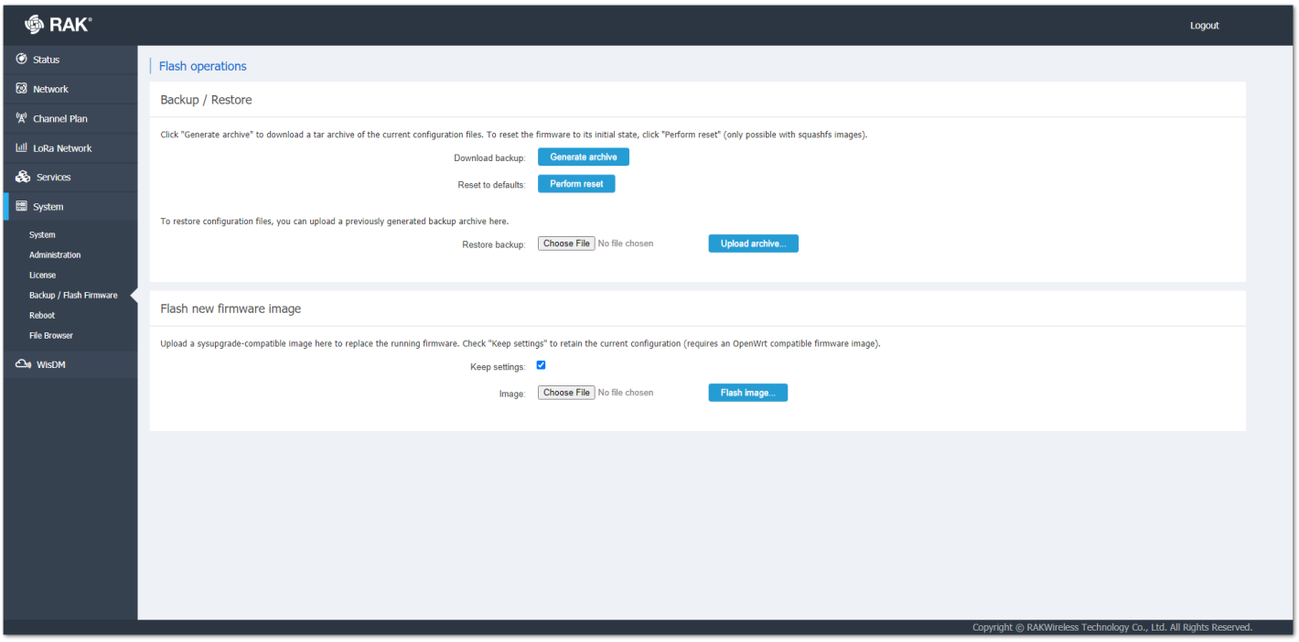

Figure 3: Upgrading Firmware

Configure Network Mode to Basic Station

1. Navigate to LoRa Network -> Network Settings.
2. change Mode in LoRaWAN Network Settings to Basic Station.
3. Select LNS Server from Server, then select TLS Server and Client Authentication from Authentication Mode.

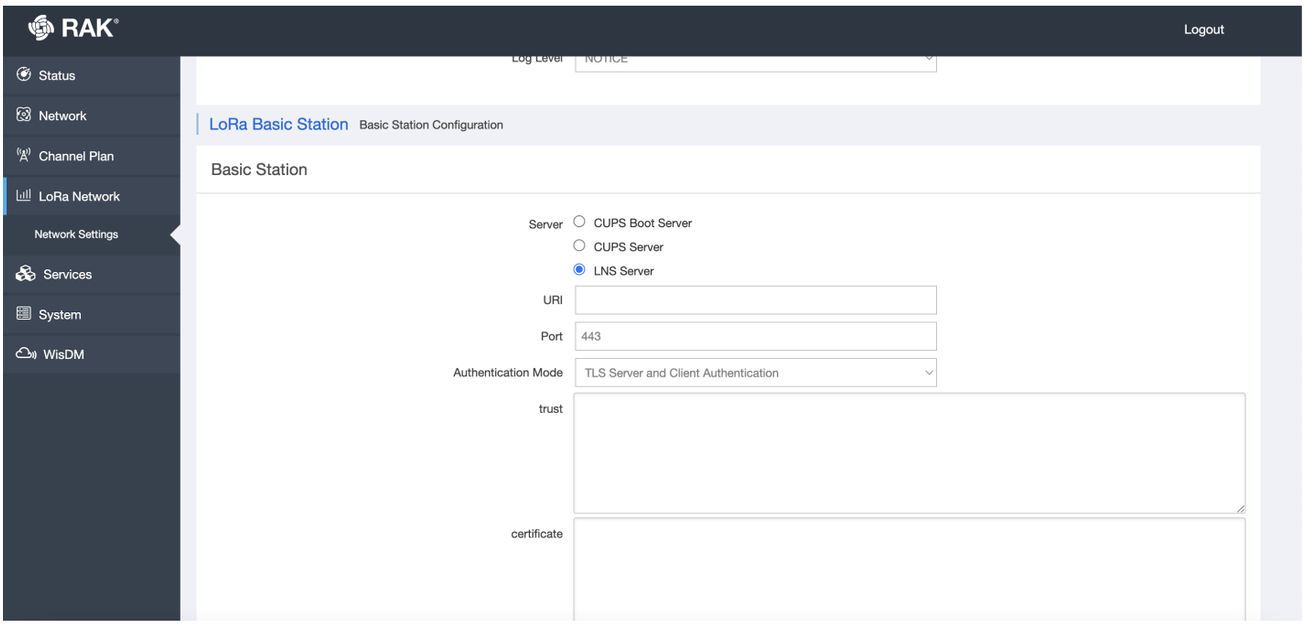

Figure 4: Configure Network Mode to Basic Station

Configure URI, Port and Authentication Mode

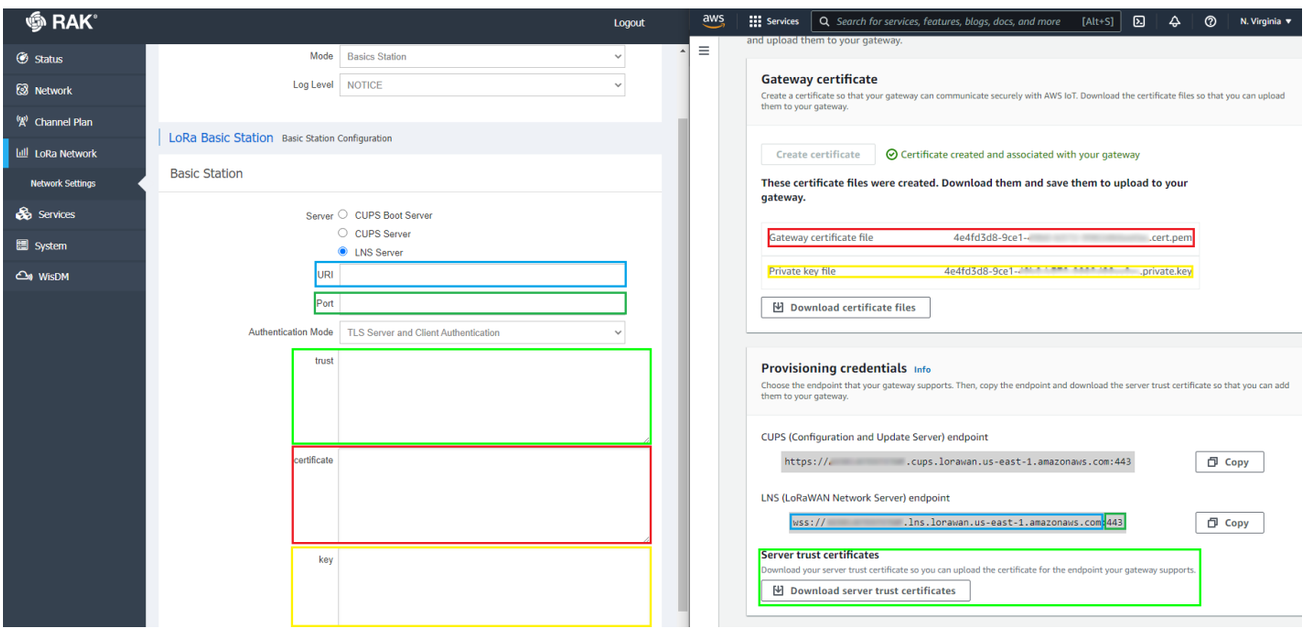

Figure 5: Configure URI, Port and Authentication Mode

Configuring the Gateway device with WisGateOS 2

Figure 6: RAK7268 With WisGate OS 2 LoRaGateway Setting

By default, the Gateway will work in Wi-Fi AP Mode which means that you can find an SSID named like "RAK7268_XXXX" on your PC's Wi-Fi Network List. "XXXX" is the last two bytes of the Gateway MAC address. To access the Web Management Platform, input the IP Address: 192.168.230.1 in your Web browser.

(Note: No password is required to connect via Wi-Fi.)

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: `<user defined>`

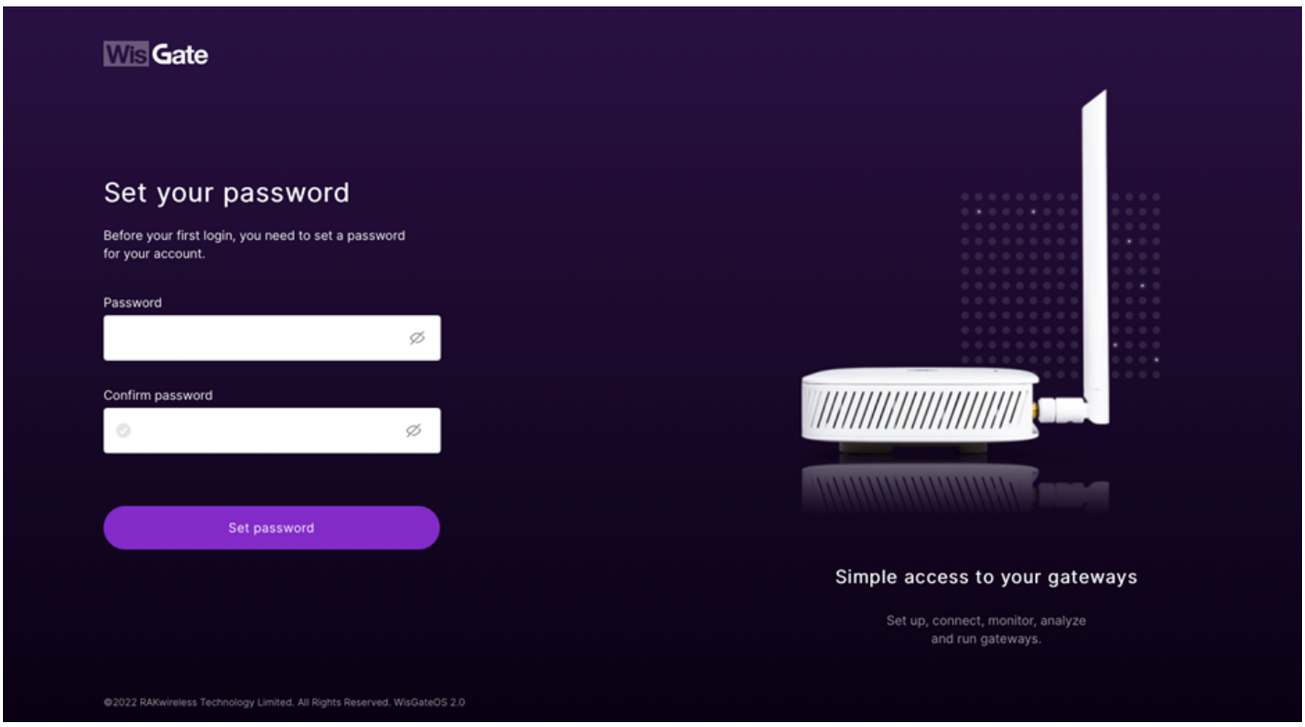

Figure 7: WisGate OS 2 Home Page

Navigate to LoRa®; change Work Mode to Basics Station and Select LNS Server from Server, then select TLS Server and Client Authentication from Authentication Mode.

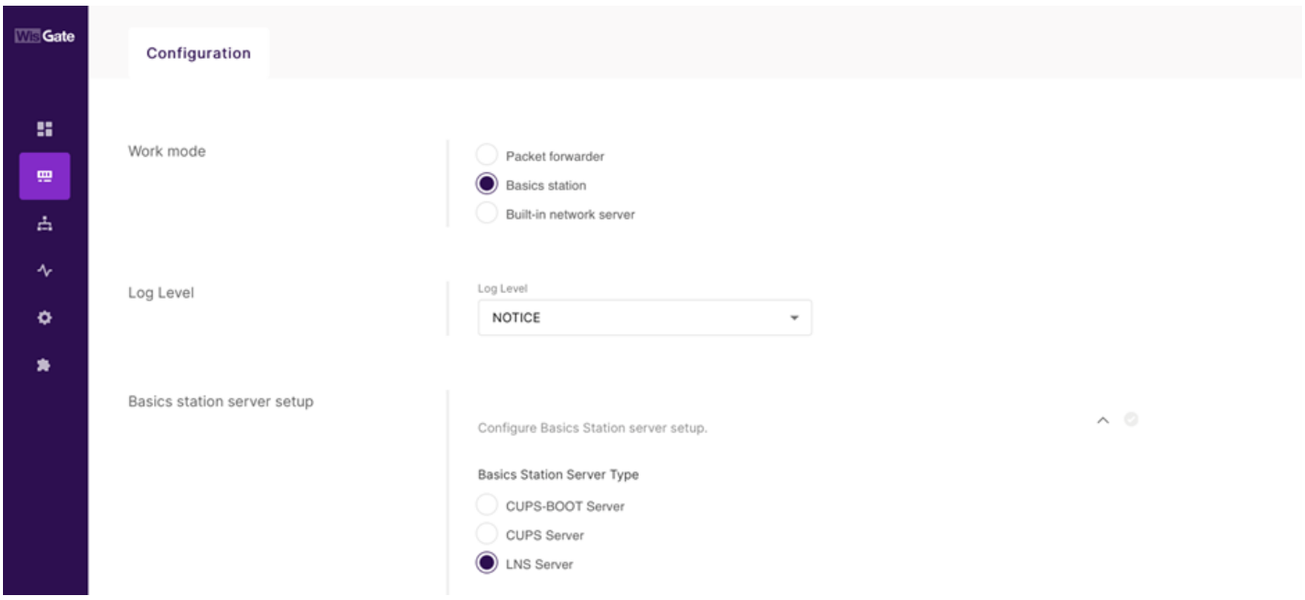

Figure 8: WisGateOS2 Basics Station Configuration

Configure URI, Port and Authentication Mode

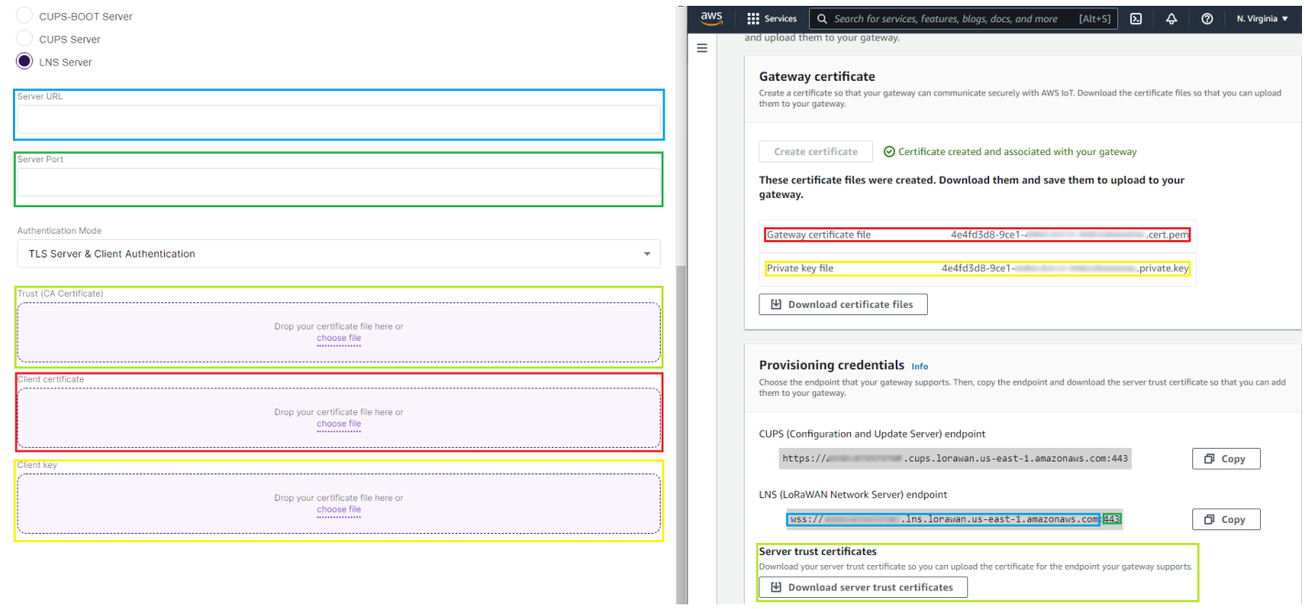

Figure 9: WisGateOS2 Basics Station Configuration

Connect the Gateway and verify the connection status

Follow the instructions in the online guide to connect your gateway to AWS IoT Core for LoRaWAN.

To verify the connection status, refer to the instructions in the section **Check gateway connection status using the console**

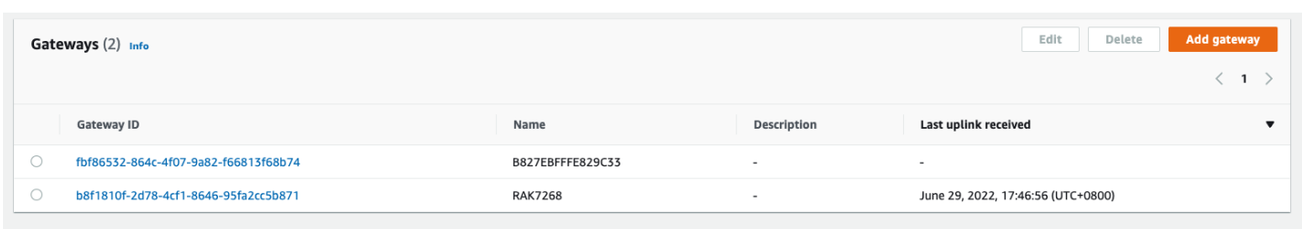

Figure 10: Gateway Connection status

Add End Device

Please refer to RAK4631 Quickstart to enable communication with the gateway.

Updating RAK4631 to RAK4631-R

RAK4631-R and RAK4631 share common hardware and are 100% identical, but they have different firmware. RAK4631-R is based on RUI3, which gives you flexibility in developing optimized firmware using the RUI3 APIs.

Please refer to updating RAK4631 to RUI3 to update RAK4631.

Join the AWS IoT LoRaWAN server

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

1. Add Device Profile

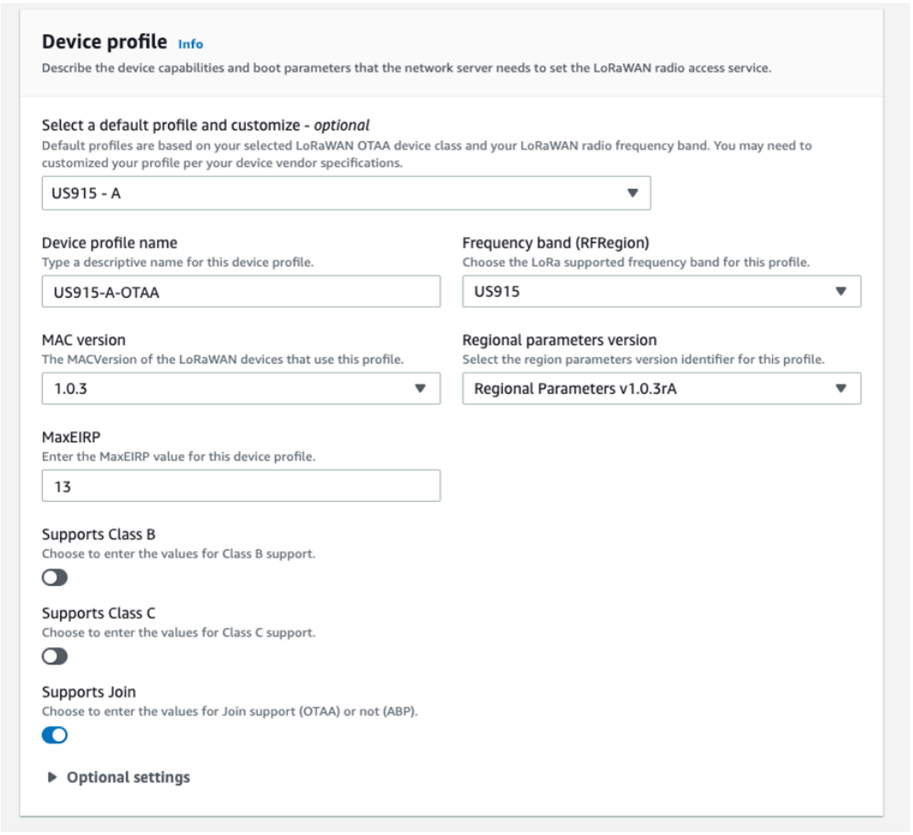

Figure 11: Adding the Device Profile

2. Add Service Profile

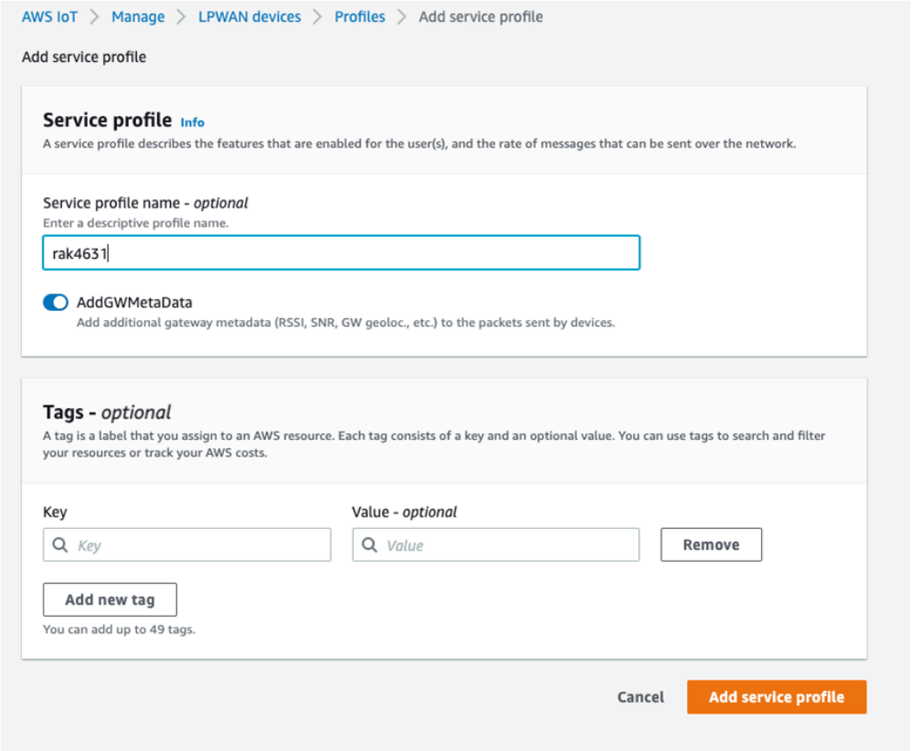

Figure 12: Adding the Service Profile

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.

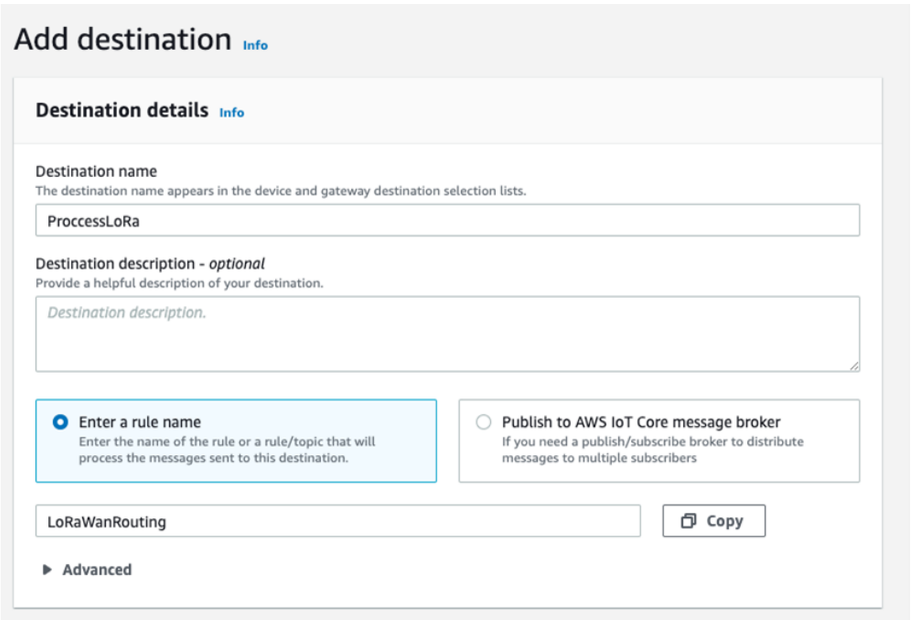

Figure 13: Adding Destination

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 to obtain the information.
  - `AT+DEVEUI`: end-device ID
  - `AT+APPEUI`: application identifier
  - `AT+APPKEY`: application key

For more AT commands, refer to the RAK4631-R AT Command Manual .

For Example:

```
AT+DEVEUI=0000000000000000
OK
AT+APPEUI=0000000000000000
OK
AT+APPKEY=00000000000000000000000000000000
OK
```

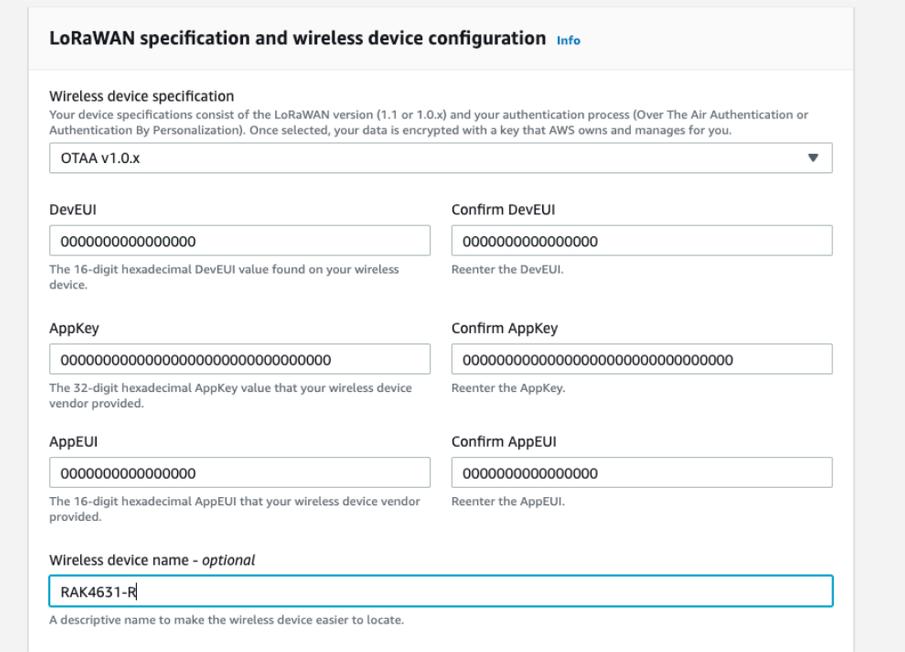

Figure 14: LoRaWAN Specifications and Wireless Device Configuration

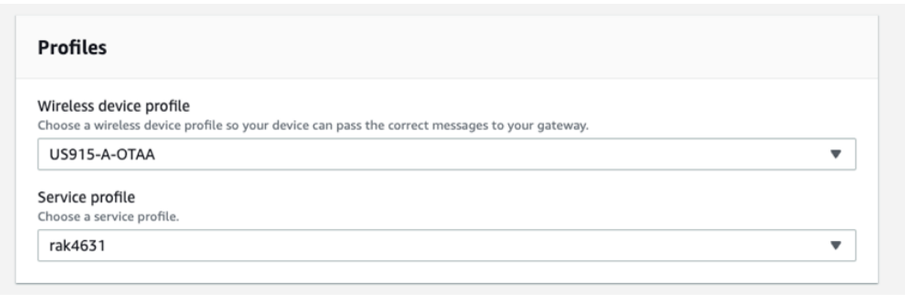

Figure 15: Choosing a Wireless Device Profile

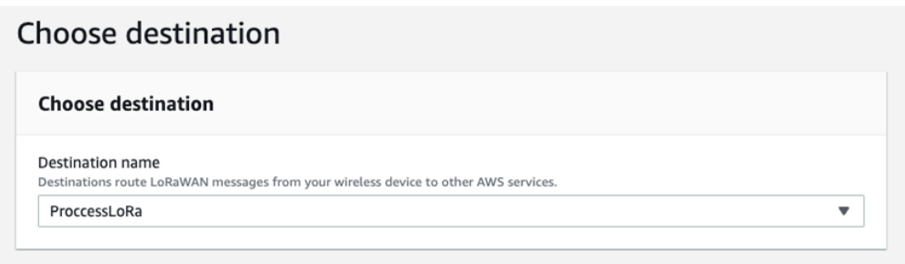

Figure 16: Choosing a Destination

5. Join the AWS IoT LoRaWAN server

Use the command: `AT+JOIN` to join the AWS IoT LoRaWAN server

```
AT+JOIN=1:0:10:8

OK
+EVT:JOINED
```

6. Send an uplink message

Use `AT+SEND` to send data on a dedicated port number

```
AT+SEND=12:112233
OK
```

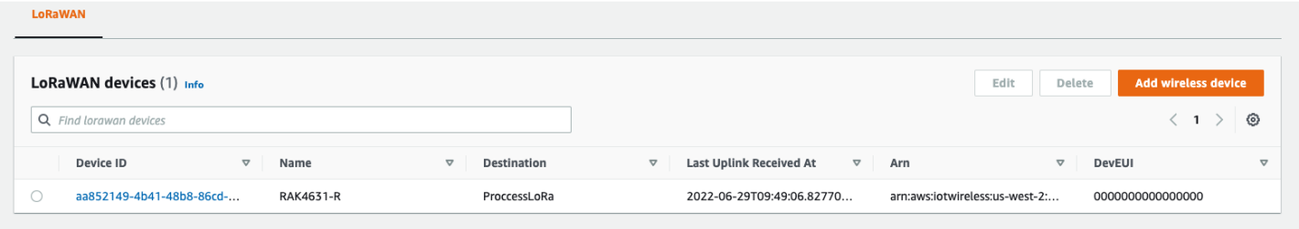

Figure 17: Uplink Received

Connect the device and verify the connection status

Follow the instructions in the online guide to connect your device to AWS IoT Core for LoRaWAN.

To verify the connection status, refer to the instructions in the section Check device connection status using the console. You can also View format of uplink messages sent from LoRaWAN devices .

Verifying Operation – a “Hello World” example

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. The architecture for this sample application is:

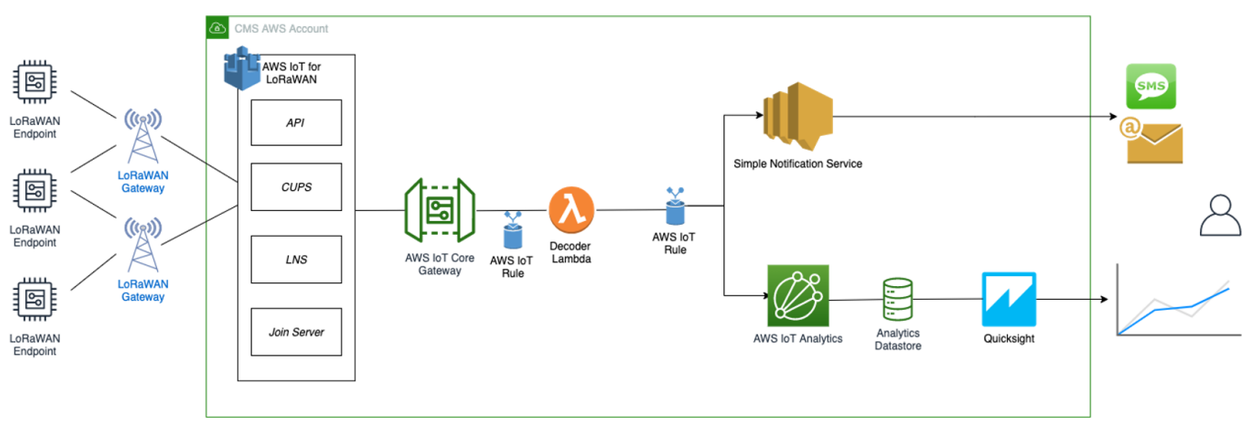

Figure 18: Sending Uplink Architecture

Create lambda function for destination rule

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

- Go to the AWS Lambda console (console.aws.amazon.com/lambda).
- Click on **Functions** in the navigation pane
- Click on **Create function**
- Select **Author from scratch**. Under Basic information, enter the function name and choose Runtime Python 3.8. from the drop-down under **Runtime**.
- Click on **Create function**.

- In the **Code** source tab, under index.js, 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/decoc
  )

  return response
```

- Once the code has been pasted, choose **Deploy** to deploy the lambda code.
- Click on the **Permissions** tab of the lambda function.
- 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.

- Return to the **Code** tab and create a test event that will allow you to test the functionality of the lambda function.
  - From the **Test** drop-down menu, 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"
        }
    }
    }
```

- Choose **Save** to save the event.
- Navigate to the AWS IoT console, choose **Test** on the navigation pane, and select **MQTT test client**.
- Configure the MQTT client to subscribe to "**#**" (all topics).
- Click on **Test** in the Lambda function page to generate the test event you just created.
- 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:

```
0000000000000088/project/sensor/decoded      February 09, 2021, 14:45:29 (UTC+0800)
{
    "devEui": "0000000000000088",
    "fPort": 1,
    "freq": "868100000",
    "timestamp": "2021-02-08T04:00:40Z",
    "temperature": -4.1
}
```
json

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, for Name, enter *LoRaWANRouting*. For **Description**, enter a description of your choice. Note the name of your rule. The information will be 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=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**.
- In **Topic ARN**, choose the topic you have created earlier.
- 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, timestamp
```

6. Under Set one or more actions, 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=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.

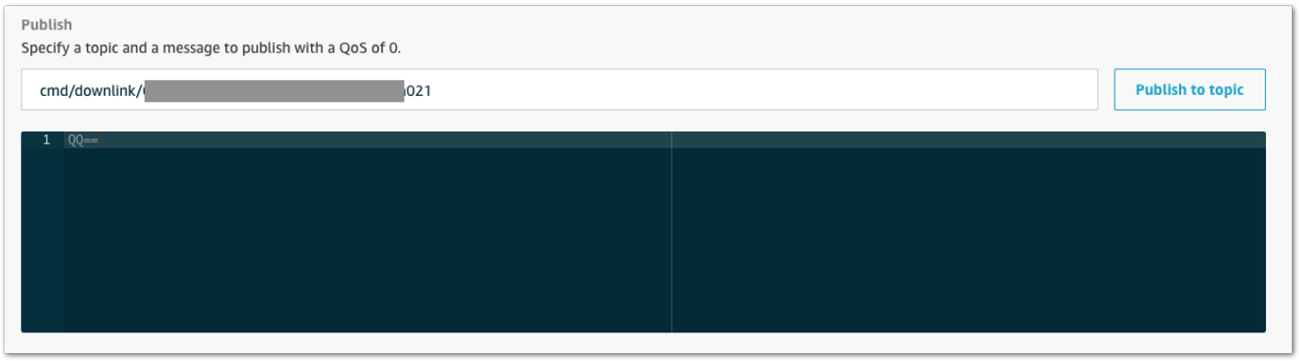

Figure 19: Specifying a Topic

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 20: 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*.
11. Choose **Quick Create**
12. Once the Quick Create Finished message is displayed, choose **Add action**.
13. 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 **Datasets**.
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.

Debugging

If you experience any issues, you can check the logs located in the /var/log/ directory.

Troubleshooting

1. Unable to see the web login:
   - Check that your wifi is connected to **RAKWireless_XXXX**.
   - Try ping **192.168.230.1**.

The Things Network v3 (TTNv3)

In this section, it will be shown how to connect RAK7268 WisGate Edge Lite 2 to TTNv3.

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

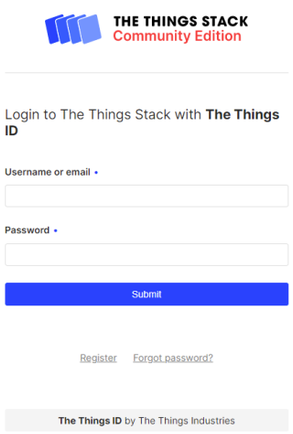

Figure 21: The Things Stack Home Page

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

Registering 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).

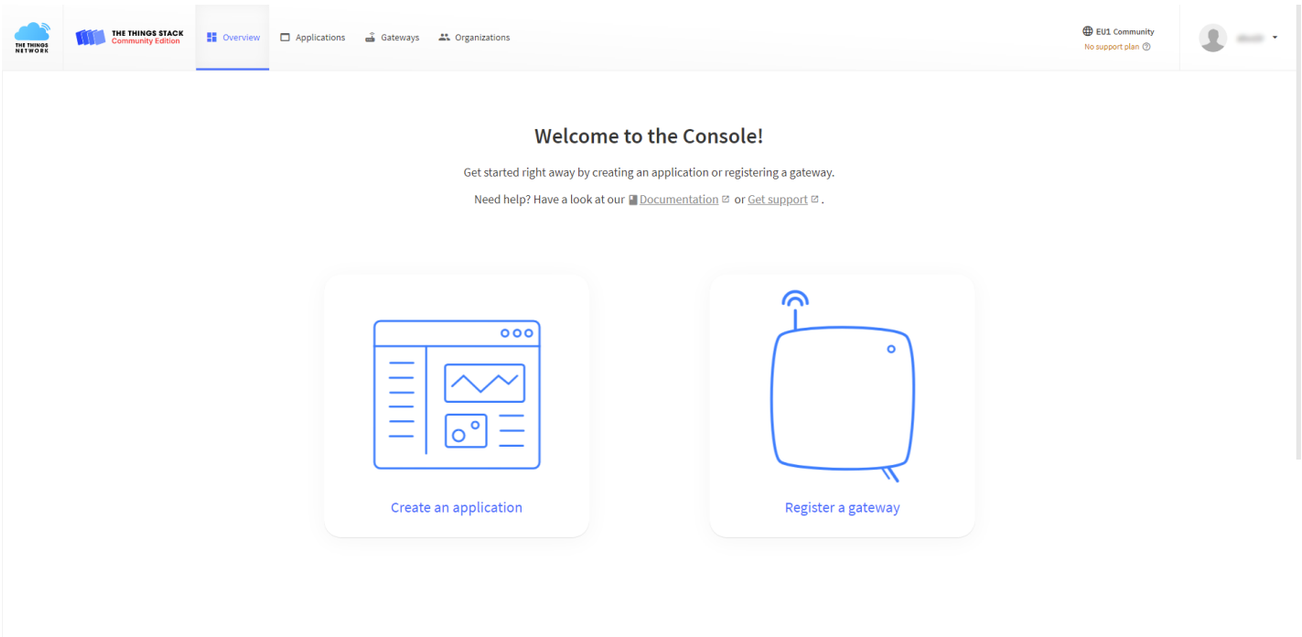

Figure 22: Console Page after successful login

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 product's Quickstart 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, we will use Europe 863-870 MHz (SF12 for RX2 - recommended).

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

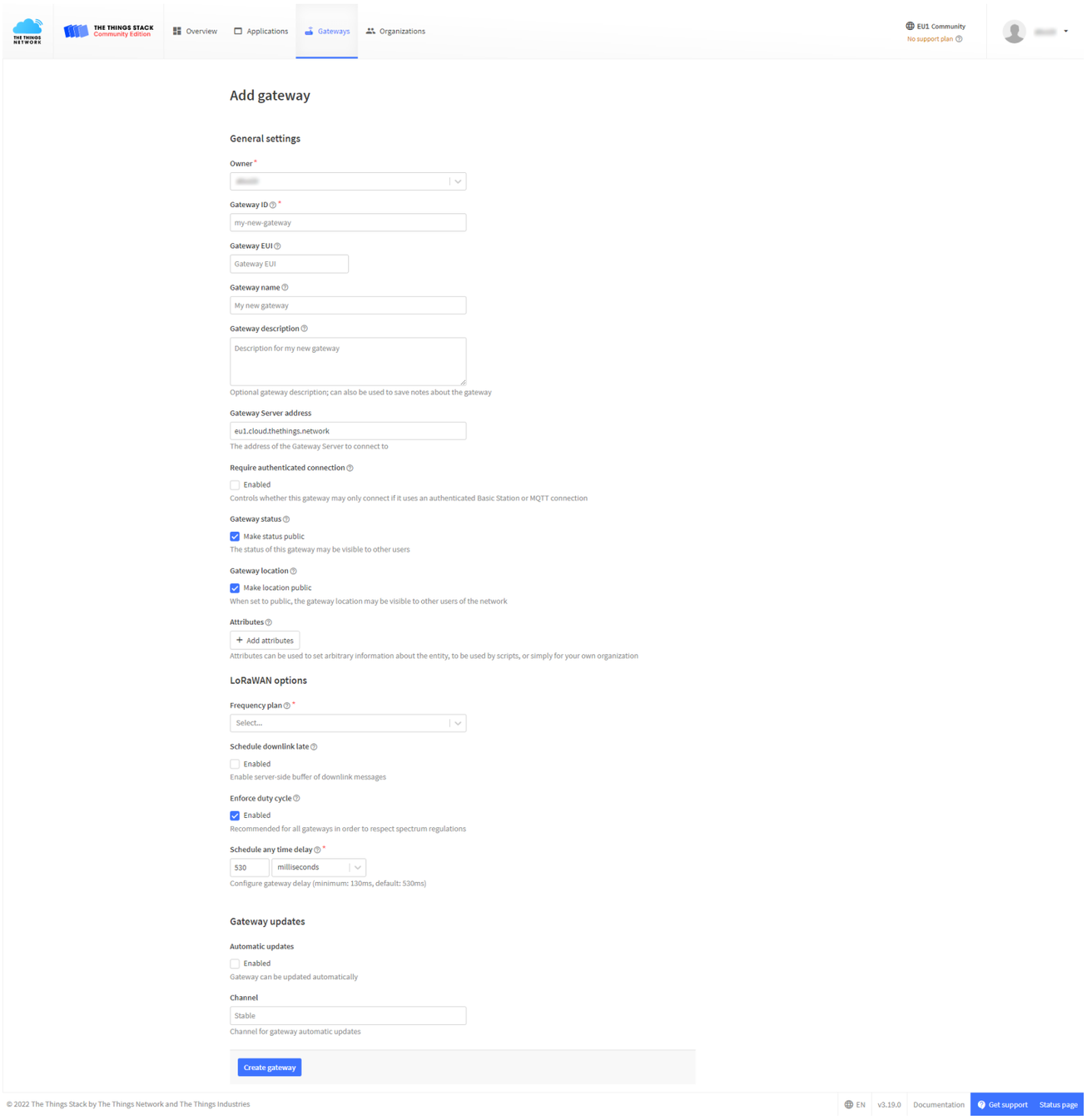

Figure 23: Adding a gateway

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

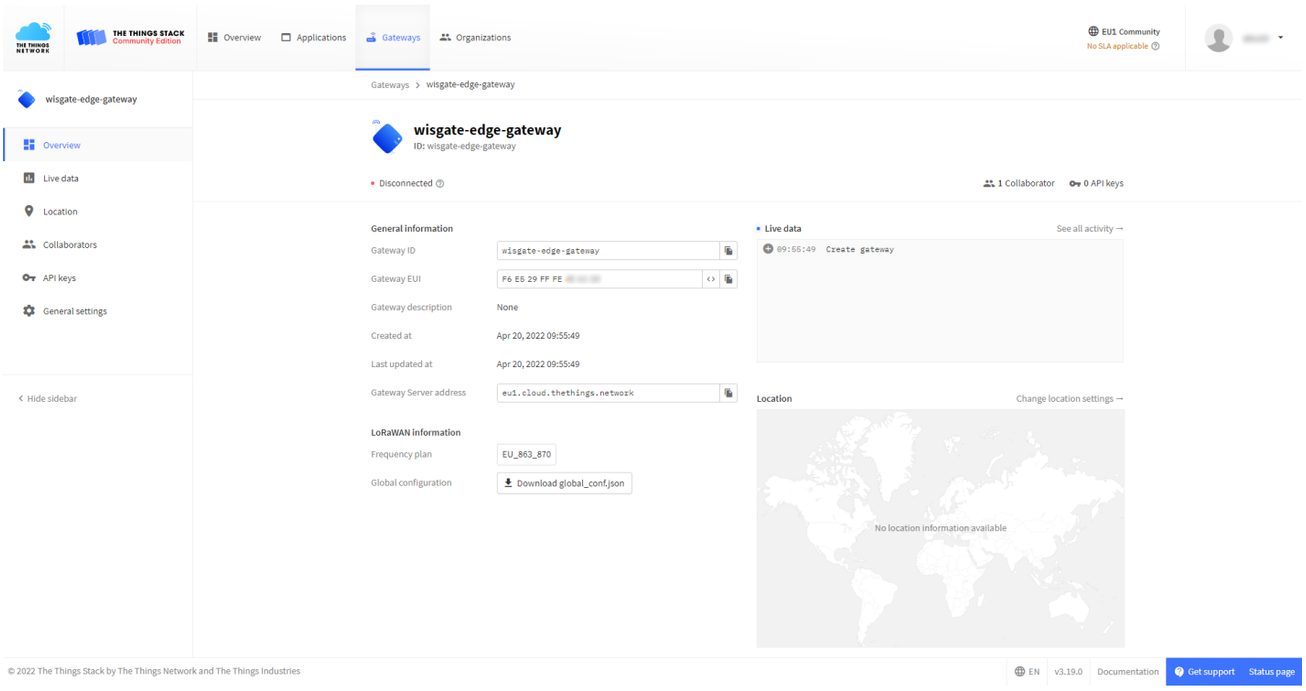

Figure 24: Overview page

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

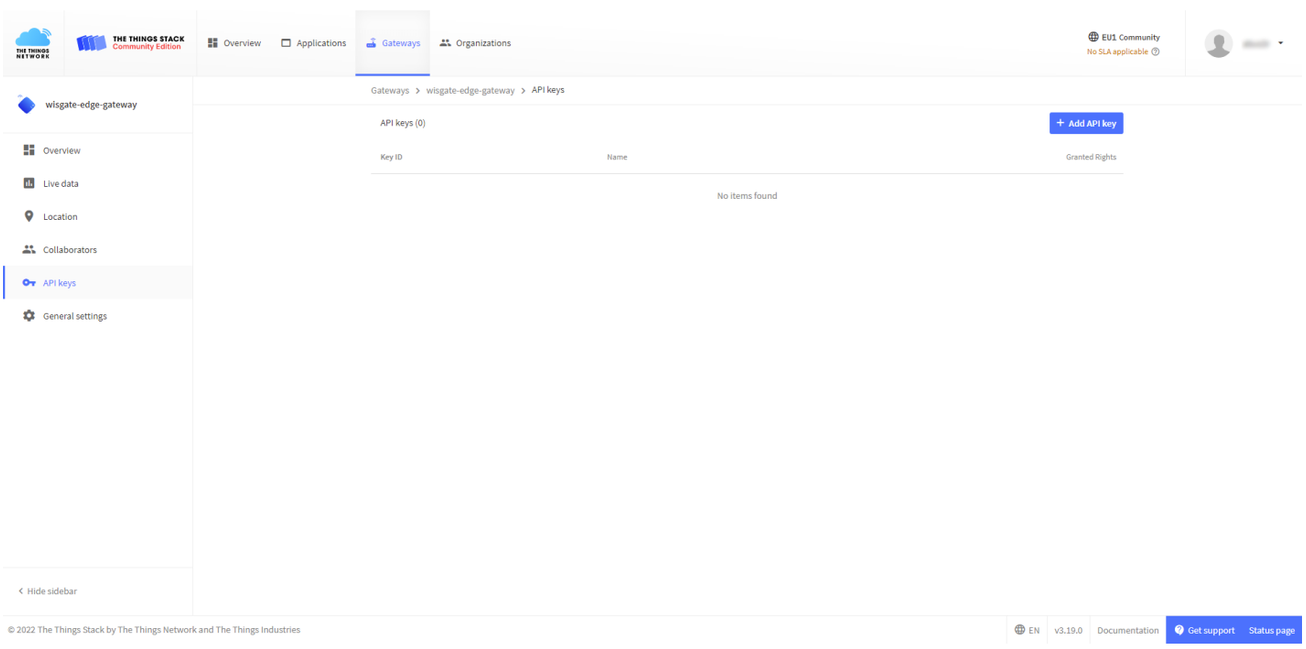

Figure 25: API key page

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**.

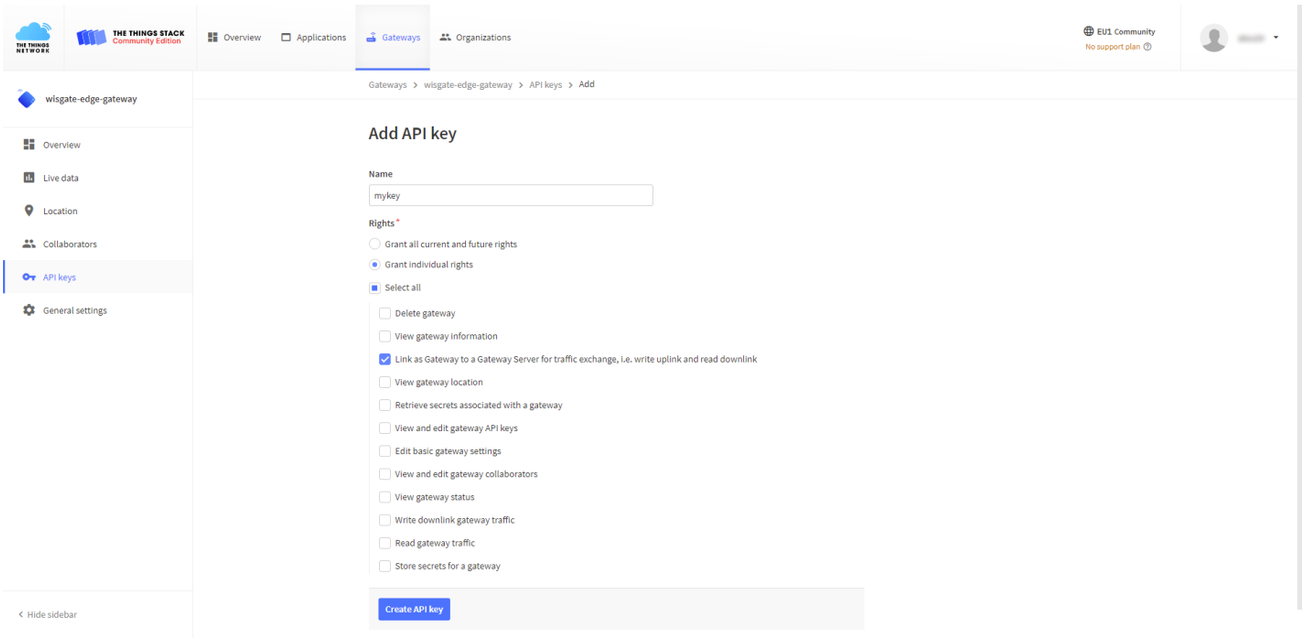

Figure 26: Generating an API key

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

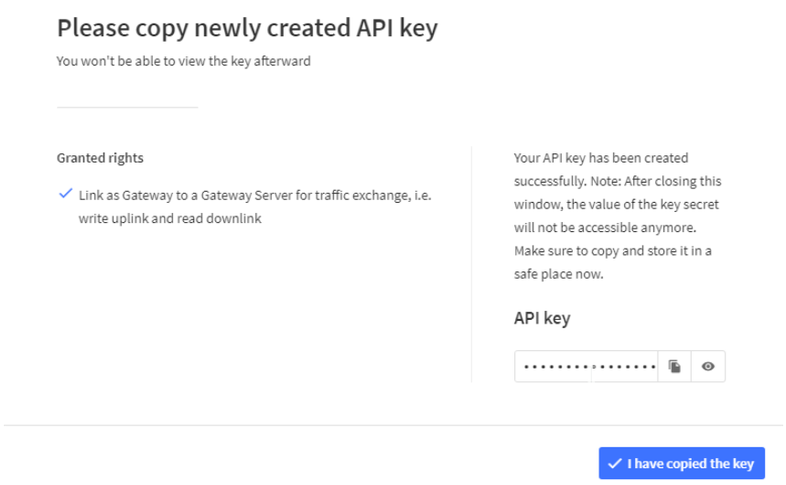

Figure 27: Copying the generated key

> ⚠ **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 check out the device's Quickstart Guide mentioned before.

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

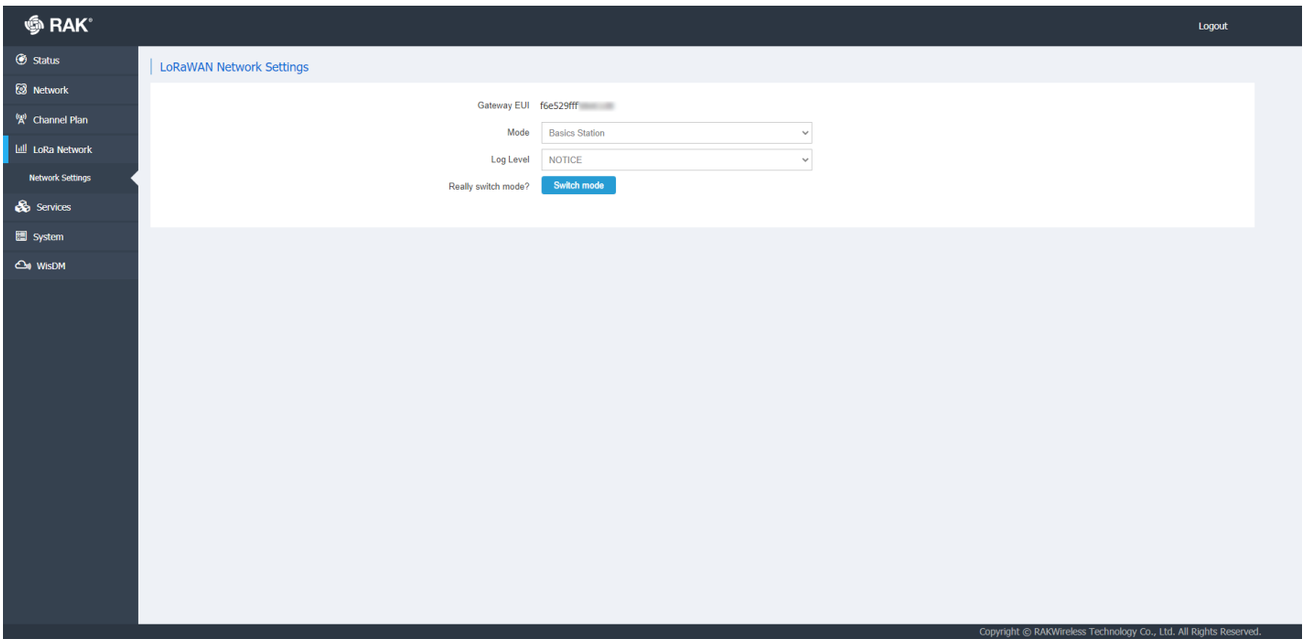

Figure 28: Changing the working mode

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 that, for this tutorial, we 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 we will use the **Let's Encrypt ISRG ROOT X1 Trust** certificate. The file with the certificate can be found here .
- **token** - This is the generated **API key**. The key must start with **Authorization:**. 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.

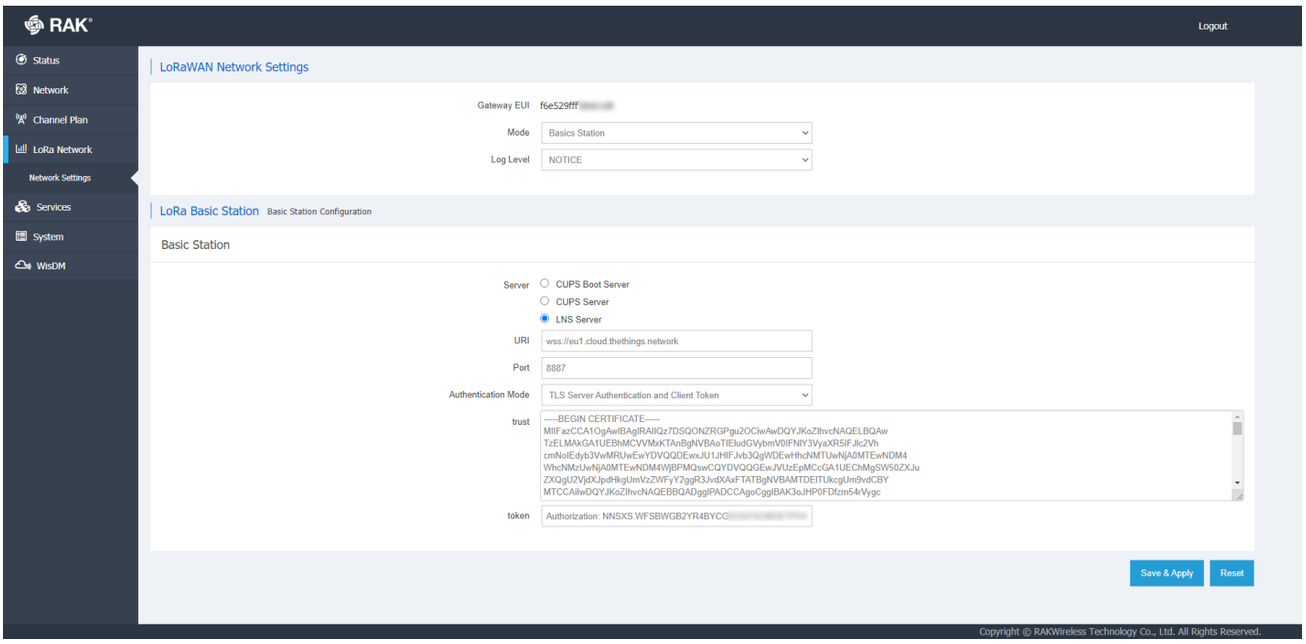

Figure 29: LoRa Basics Station settings

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

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

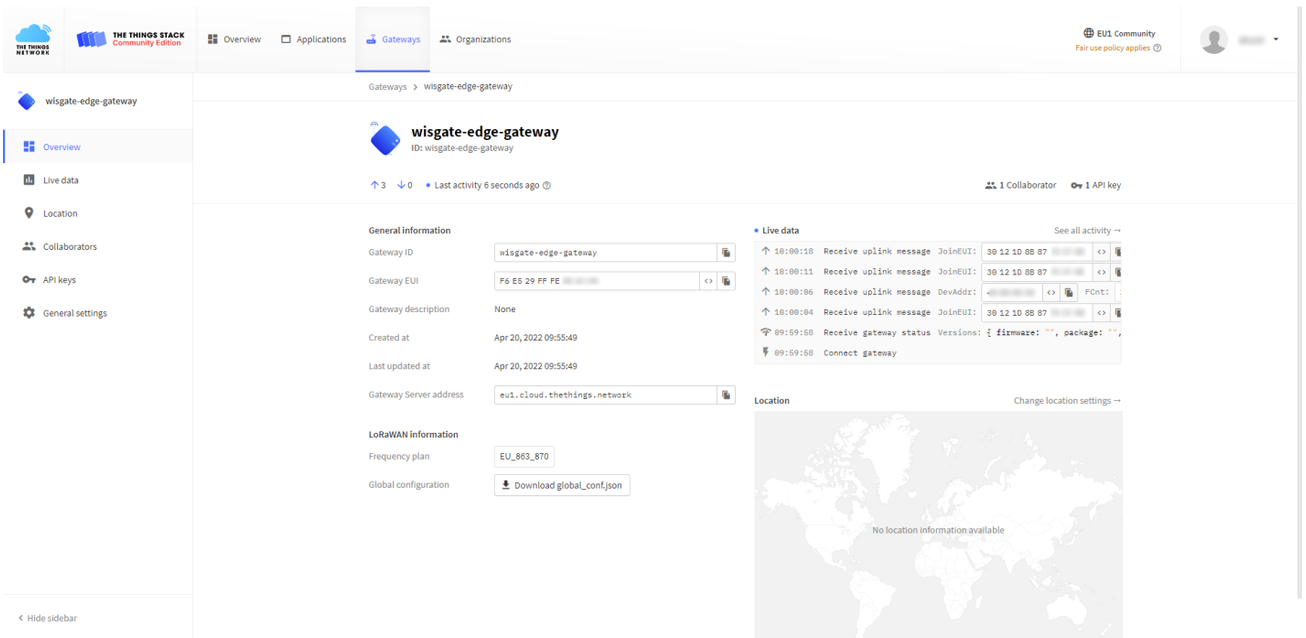

Figure 30: Successful connection

LORIOT

In this tutorial, you will learn how to connect RAK7268 WisGate Edge Lite 2 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

- RAK7268 WisGate Edge Lite 2

Software

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

Registering the Gateway

1. Log into your LORIOT account.

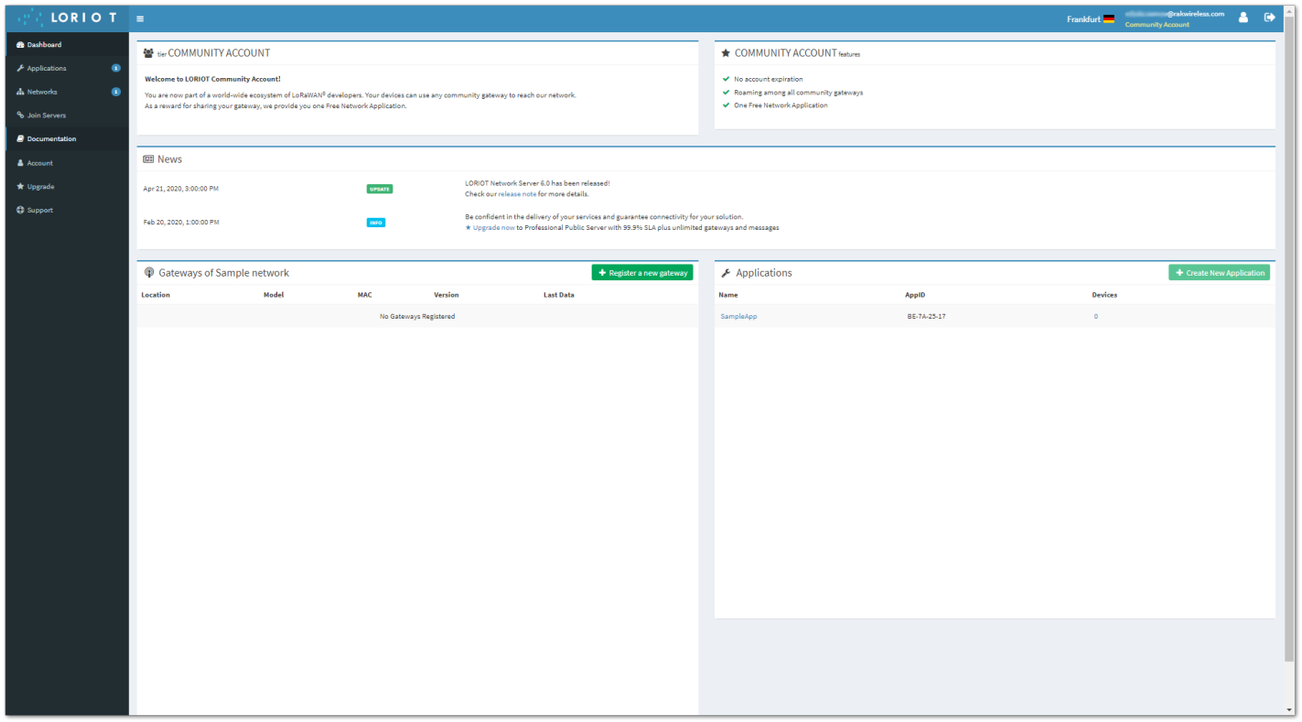

Figure 31: LORIOT Homepage

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**.

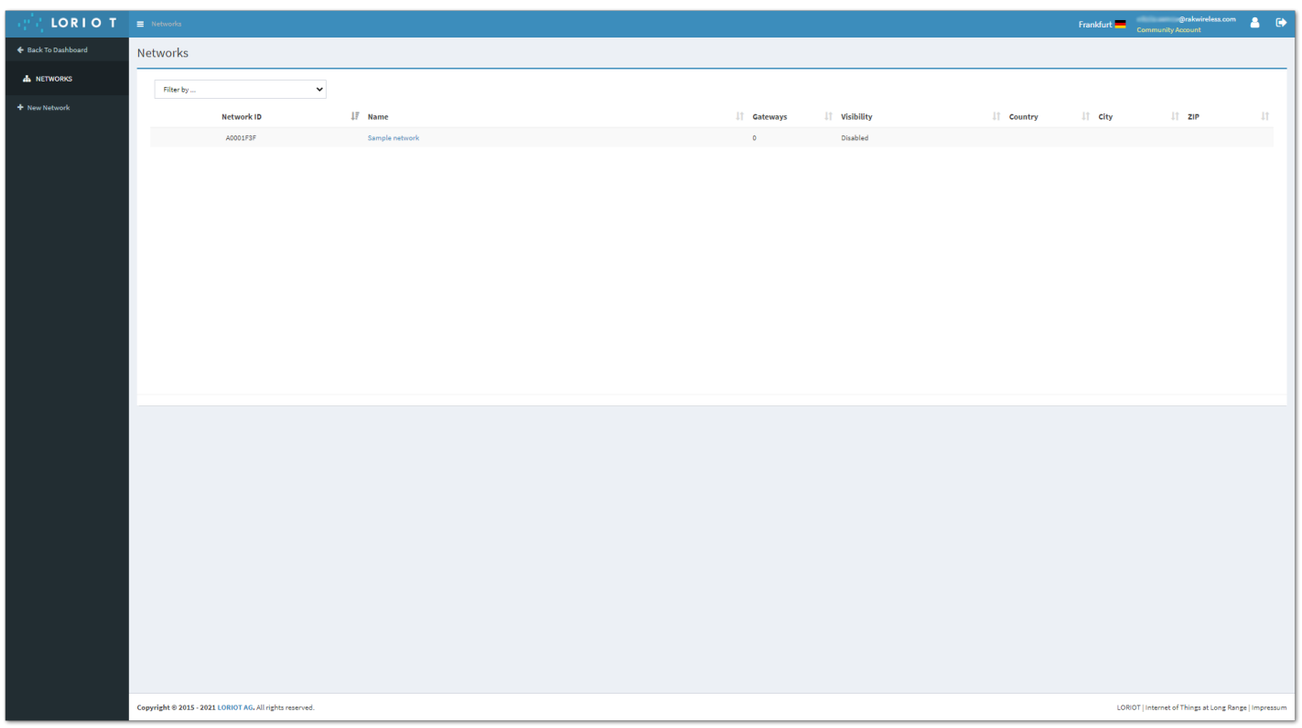

Figure 32: Networks List

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

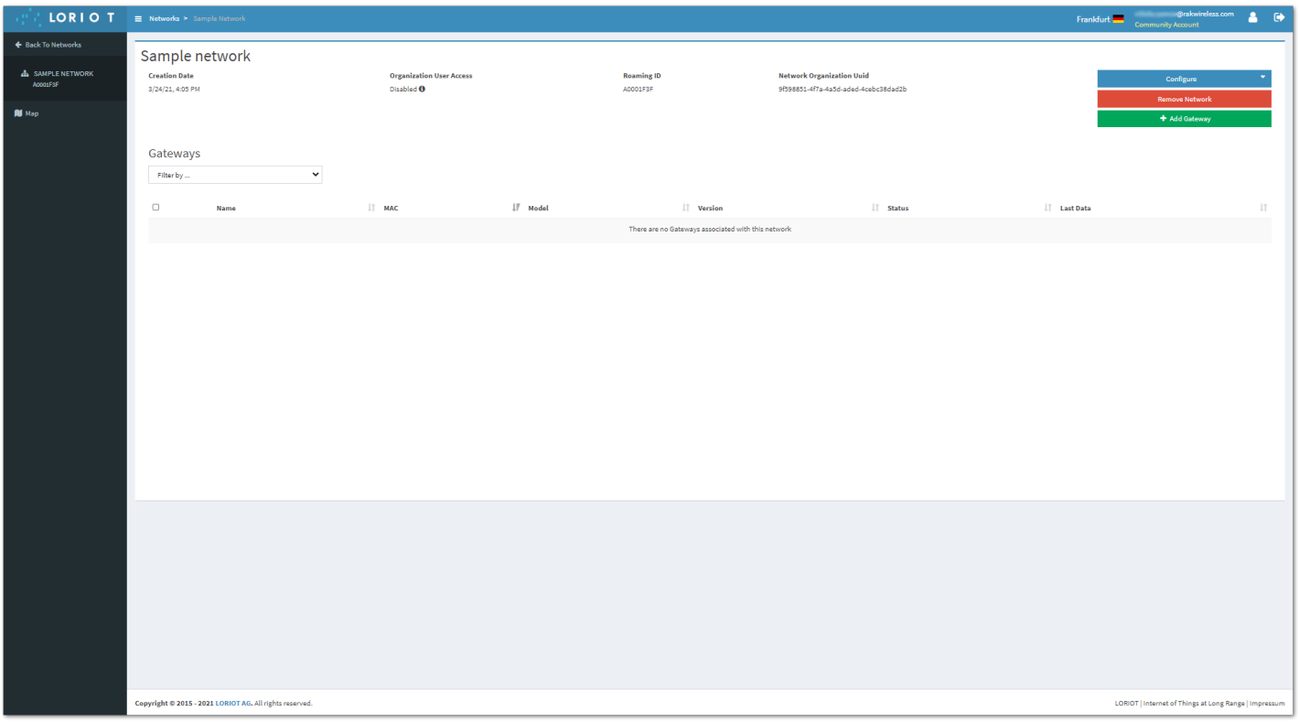

Figure 33: Adding a gateway to the network

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.

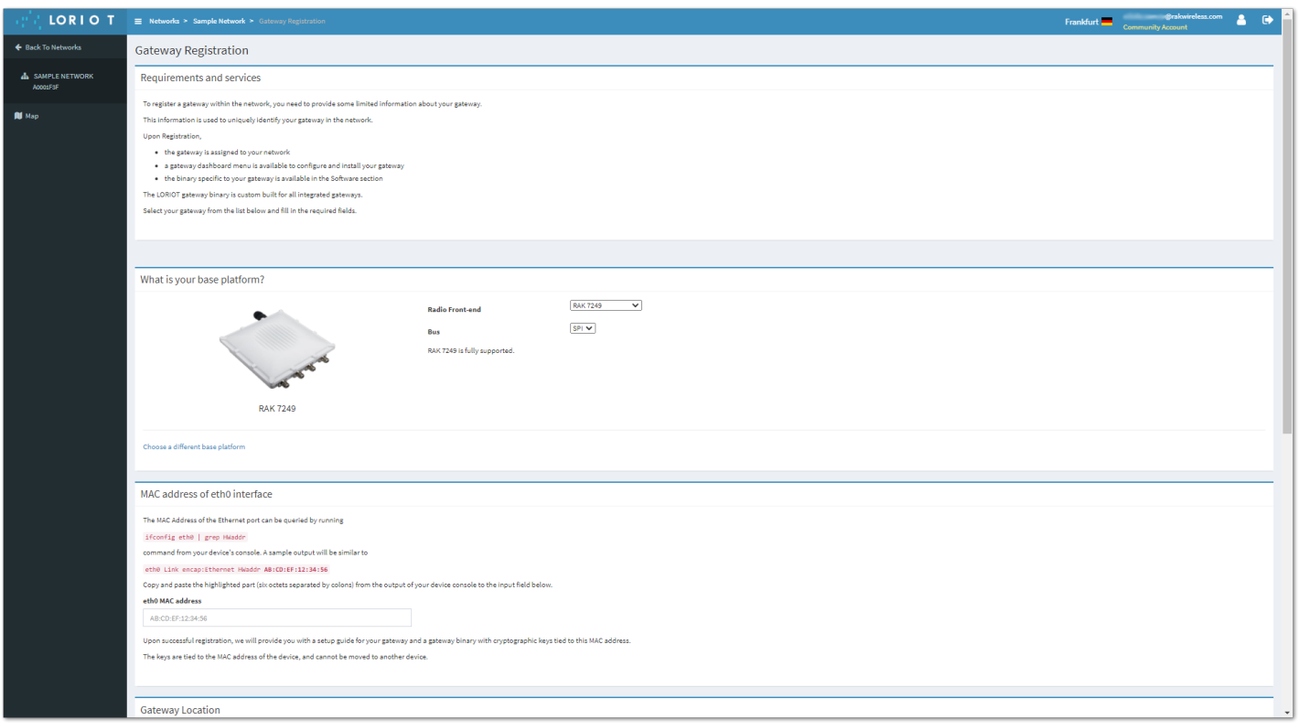

Figure 34: Selecting RAK7249

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**.

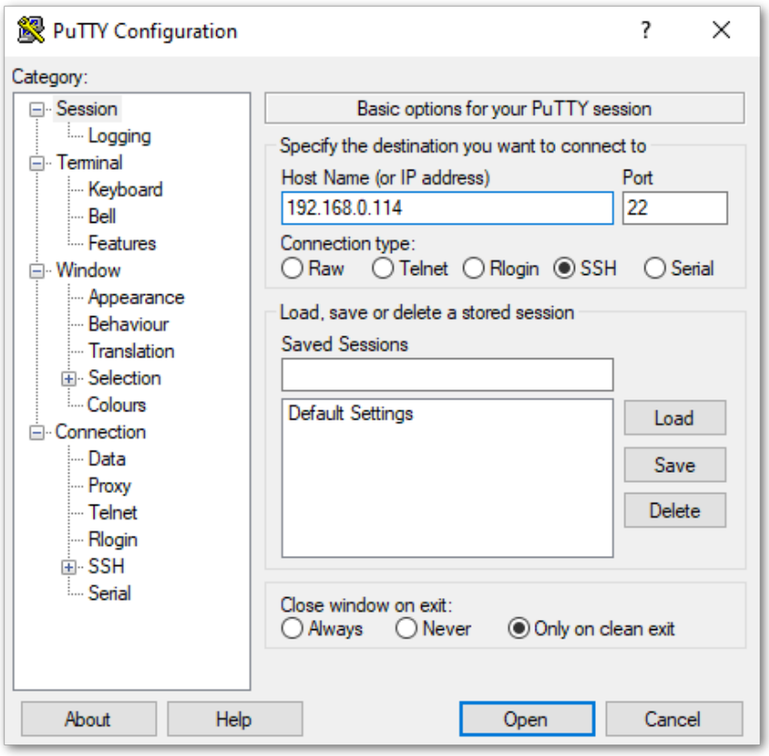

Figure 35: PuTTY Configuration

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
```

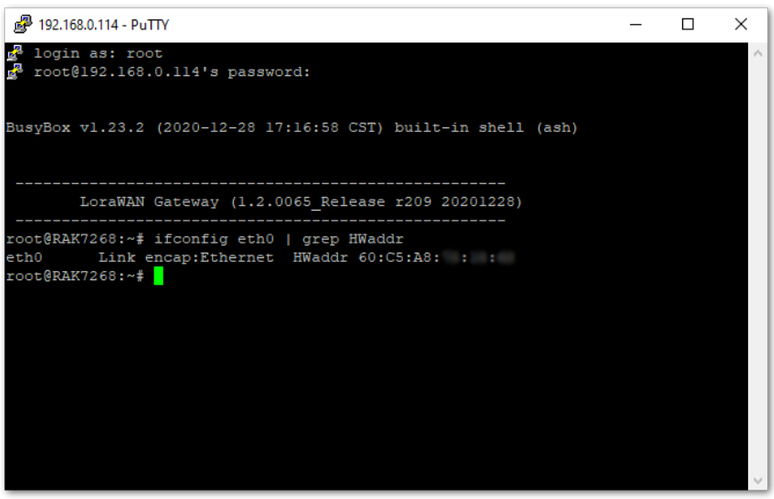

Figure 36: Getting the MAC address of the gateway

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.

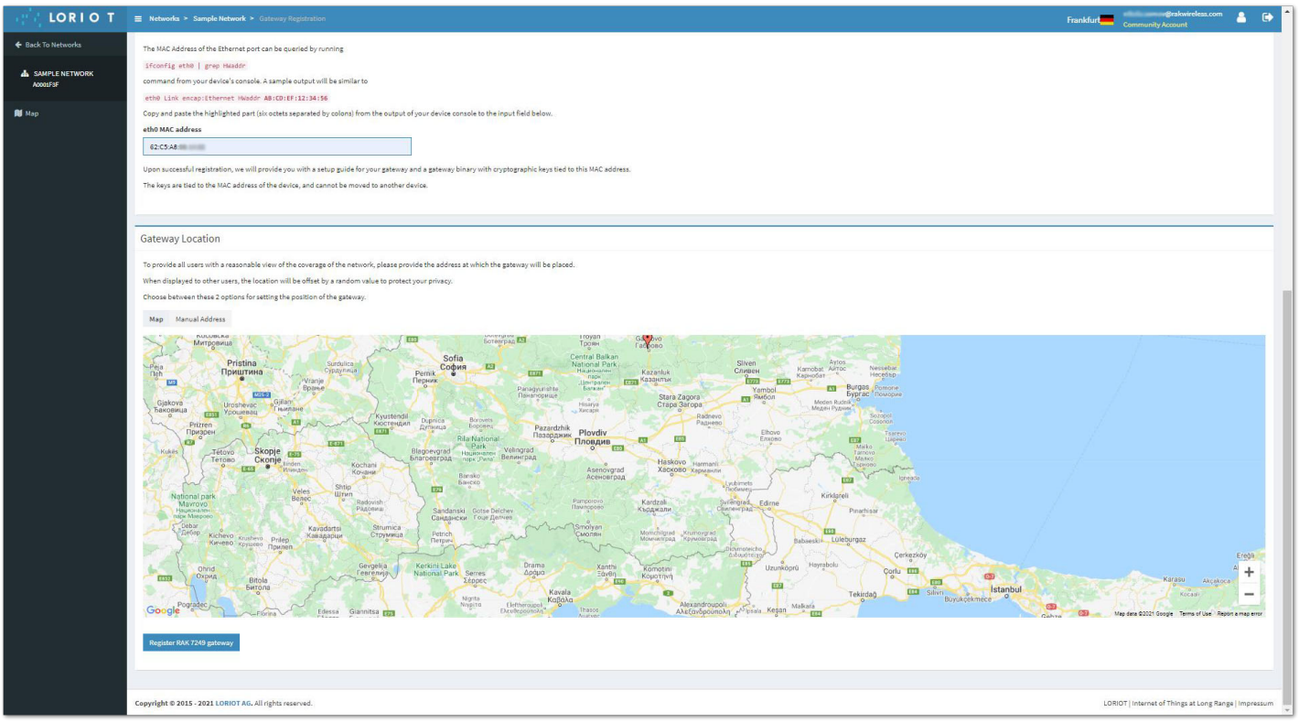

Figure 37: Filling out the MAC address

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/sx130x_lora_pkt_fwd disable; /etc/init.d/loriot-gw enable; reboot now
```

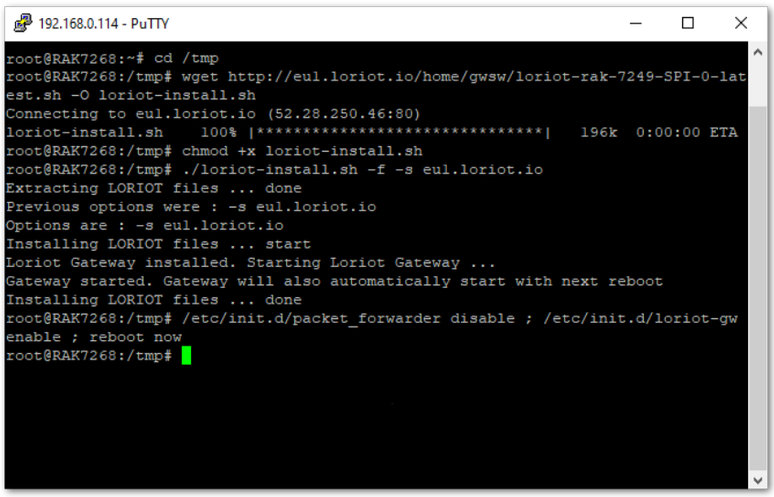

Figure 38: Installing LORIOT software

Your gateway is now registered and connected to LORIOT.

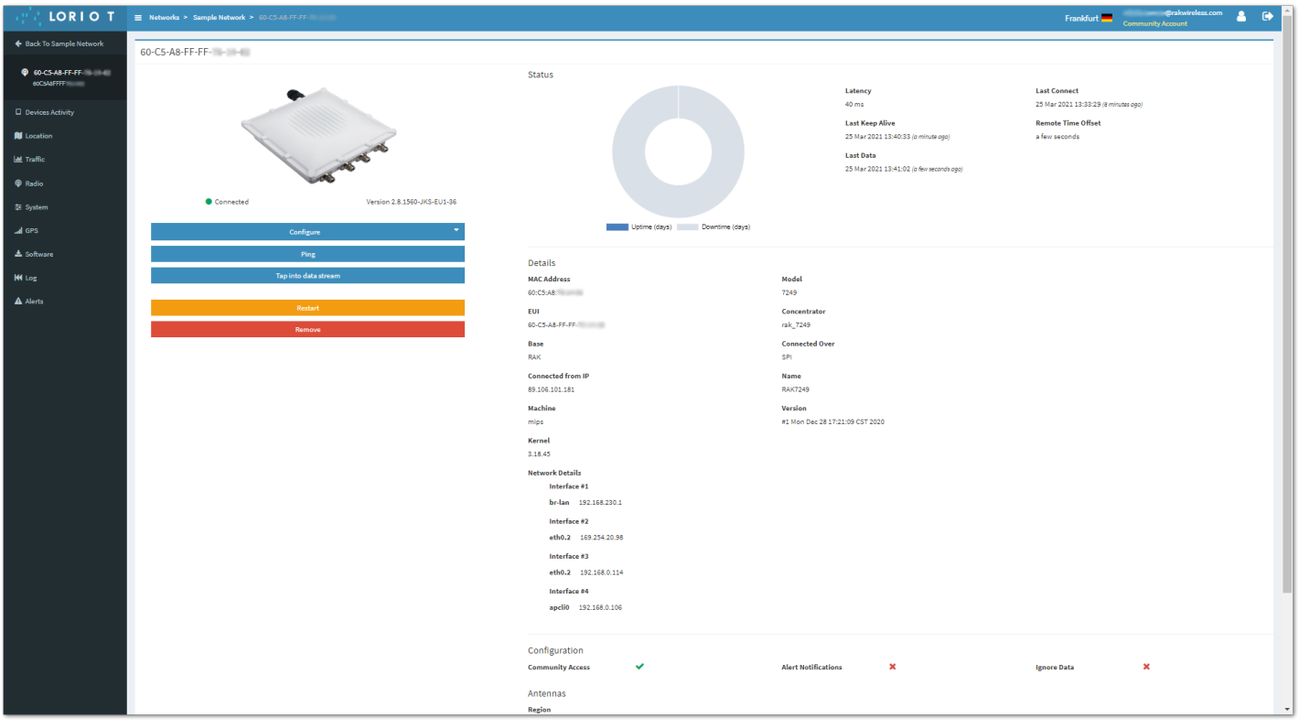

Figure 39: Successful Connection

Last Updated: 7/29/2022, 10:17:19 PM