IoT sensor dashboard

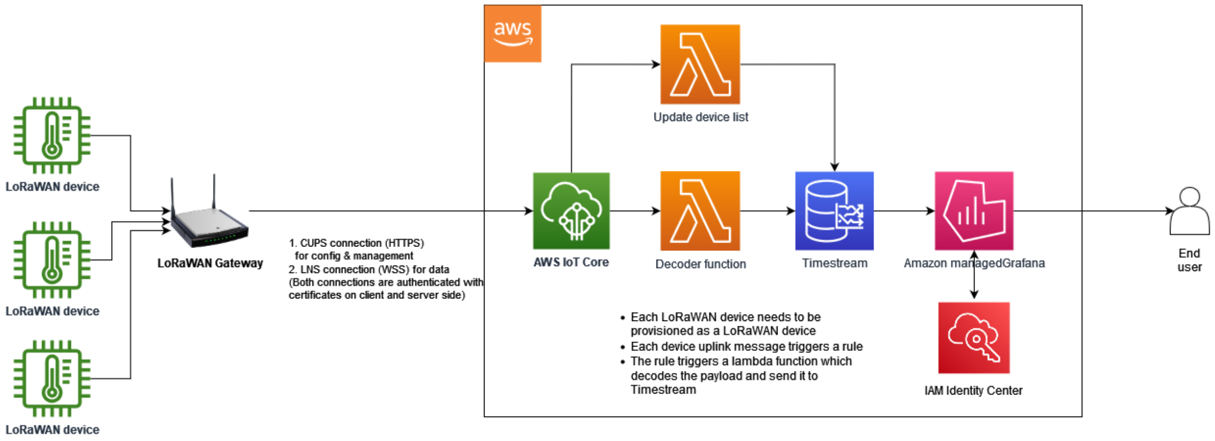
User Guide

[Author name]
7-28-2023

1. Introduction

This is the user guide for the IoT sensor dashboard accelerator. It should be noted that since this is an accelerator, it is designed to be a starting point of your solution build journey rather than a complete solution. This user guide contains the important information related to the architecture, AWS services utilized, and instructions on how to provision gateways and sensors. This accelerator has to be deployed in your AWS account prior to any provisioning of hardware. Refer the Deployment Guide for more details and step by step instructions on the deployment.

2. Architecture



1. CUPS connection (HTTPS)
for config & management
2. LNS connection (WSS) for data
(Both connections are authenticated with certificates on client and server side)

- Each LoRaWAN device needs to be provisioned as a LoRaWAN device
- Each device uplink message triggers a rule
- The rule triggers a lambda function which decodes the payload and send it to Timestream

3. Functional description

3.1 AWS IoT Core

AWS IoT Core lets you connect billions of IoT devices and route trillions of messages to AWS services without managing infrastructure.

There are 2 key capabilities of the AWS IoT Core service utilized for this accelerator

- AWS IoT Core – This acts as the connection point to ingest sensor data directly sent from MQTT sensors. LoRaWAN sensors utilize the AWS IoT Core for LoRaWAN capability.
  AWS IoT Core for LoRaWAN is a fully managed LoRaWAN Network Server (LNS) that enables customers to connect wireless devices that use the LoRaWAN protocol for low-power, long-range wide area network connectivity with the AWS Cloud. Using AWS IoT Core, customers can now set up a private LoRaWAN network by connecting their LoRaWAN devices and gateways to the AWS Cloud—without the need to develop, maintain, or operate their own LoRaWAN Network Server.
  The sensors and gateways for the LoRaWAN sensor network is provisioned within the IoT Core for LoRaWAN. All the sensors that are provisioned need to have tags to identify the vendor and the model of the sensor which are used in the decoder lambda function as described in the next section.
  Refer the sensor and gateway provisioning section below for step by step instructions on provisioning.
  The following key resources are created as part of the accelerator:
    - **Device profiles** - Device profiles define the device capabilities and boot parameters that the network server uses to set the LoRaWAN radio access service. It includes selection of parameters such as LoRa frequency band, LoRa regional parameters version, and MAC version of the device.
    - **Default Service profile** - Service profiles describe the communication parameters the device needs to communicate with the application server.
    - **Default destination** - destinations describe the AWS IoT rule that processes a device's data for use by AWS service

- AWS IoT Core rules engine - AWS IoT rules send device messages to other services. AWS IoT rules can also process the binary messages received from a LoRaWAN device to convert the messages to other formats that can make them easier for other services to use.
  AWS IoT Core for LoRaWAN destinations associate a wireless device with the rule that processes the device's message data to send to other services. The rule acts on the device's data as soon as AWS IoT Core for LoRaWAN receives it. AWS IoT Core for LoRaWAN destinations can be shared by all devices whose messages have the same data format and that send their data to the same service.
    - **Default IoT Rule** – in this accelerator, this rule triggers the decoder Lambda function described in the next section
    - **Thing Event Rule** – this rule is triggered whenever a thing is created or deleted in IoT Core
    - **Generic Sensor Rule** – this rule is triggered when a MQTT sensor published directly to a topic of the format *<devEui>/tx*

3.2 AWS Lambda

AWS Lambda is a compute service that lets you run code without provisioning or managing servers. Lambda runs your code on a high-availability compute infrastructure and performs all of the administration of the compute resources, including server and operating system maintenance, capacity provisioning and automatic scaling, and logging. With Lambda, all you need to do is supply your code in one of the language runtimes that Lambda supports.

In this accelerator, there are 2 Lambda functions:

- **Decoder Lambda**- This is a key part of this accelerator. This function is triggered on every uplink message from a sensor and does the following:
    - Lookup the vendor and model of the sensor based on the 'Vendor' and 'Model' tags
    - Decode the binary payload sent by the sensor by invoking the relevant decoder function for the vendor and model
    - Write the decoded data into Amazon Timestream
- **Update device list Lambda**- This function is triggered whenever a thing is created or deleted from the device registry in IoT Core. When a LPWAN device is provisioned in IoT Core, it auto provisions a thing in the device registry. IoT Core has an inbuilt feature to publish such device related events to certain predefined topics in its message broker. The Thing Event Rule described in the previous section, receives any such events published to these topics and triggers this function. On getting triggered, it writes the event to a table (separate from the timeseries sensor data table) in Timestream. Refer the next section for more details on the Timestream tables.

3.3 Amazon Timestream

Amazon Timestream is a fast, scalable, and serverless time-series database service that makes it easier to store and analyze trillions of events per day up to 1,000 times faster. Amazon Timestream automatically scales up or down to adjust capacity and performance, so that you don't have to manage the underlying infrastructure.

There are 2 tables in the Timestream database which are created by this accelerator:

- **sensor_data** – this contains the timeseries values sent by the sensors.
    - Dimensions -
        - DevEui
    - measure_name – depends on the quantity measured by the sensor. A sensor sending multiple quantities per each uplink will cause multiple writes, one row per each quantity
    - measure_value – this can be any value based on the value type returned by the decoder function

- **sensor_list** – this contains the information about the sensors that were created or deleted along with the timestamps in the form of an audit log

- Dimensions –
  - DevEui – eg: 0102030405060708
  - Vendor – eg: Netvox
  - Model – eg: R718A
  - Name – friendly name for the sensor. Eg: Room TH Sensor 1
- measure_name – “status”
- measure_value – “CREATED” or “DELETED”

3.4 Amazon Managed Grafana

Amazon Managed Grafana is a fully managed service for Grafana, a popular open-source analytics platform that enables you to query, visualize, and alert on your metrics, logs, and traces. This accelerator utilizes Amazon Managed Grafana (AMG) for visualizing the data sent by the sensors.

This accelerator, during the deployment, executes the following tasks:

- Create an AMG workspace
- Create a Timestream datasource
- Load template dashboards and update the data source uids of the templates with the uid of the Timestream datasource created in previous step

This ensures that the template dashboards are useable right out of the box. The sample sensor dashboards folder contains template dashboards for each type of sensor (vendor & model). These templates contain a dropdown menu which displays the list of sensors of the same type that are provisioned and allows the user to view the data from a selected sensor – both current and historical.

Since dashboard building in Grafana is a no code/low code exercise, these templates can be used as a starting point to develop more comprehensive dashboarding features to cater to more specific customer requirements.

3.5 AWS IAM Identity Center

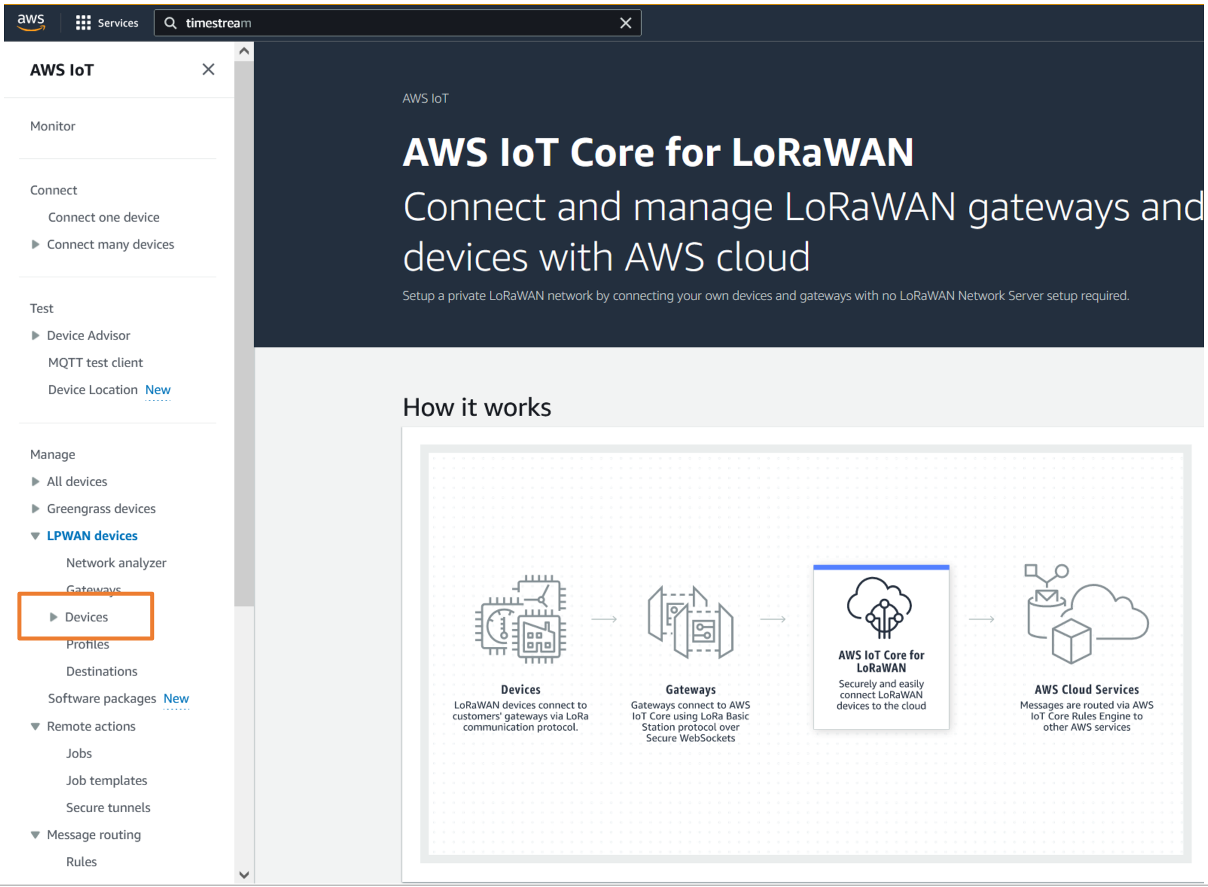
WS IAM Identity Center (successor to AWS Single Sign-On) helps you securely create or connect your workforce identities and manage their access centrally across AWS accounts and applications. IAM Identity Center is the recommended approach for workforce authentication and authorization on AWS for organizations of any size and type. Using IAM Identity Center, you can create and manage user identities in AWS, or connect your existing identity source, including Microsoft Active Directory, Okta, Ping Identity, JumpCloud, Google Workspace, and Azure Active Directory (Azure AD).

This accelerator utilizes the IAM Identity Center as the SSO for logging into the Grafana dashboards. Users defined in IAM Identity Center can be assigned with desired privileges as viewer/editor/admin from the AMG page in the AWS console.

4. Provisioning guide

4.1 Provision a LoRaWAN device/sensor

4.1.1 Navigate to IoT Core > LPWAN Devices > Devices



AWS IoT Core for LoRaWAN

Connect and manage LoRaWAN gateways and devices with AWS cloud

Setup a private LoRaWAN network by connecting your own devices and gateways with no LoRaWAN Network Server setup required.

How it works

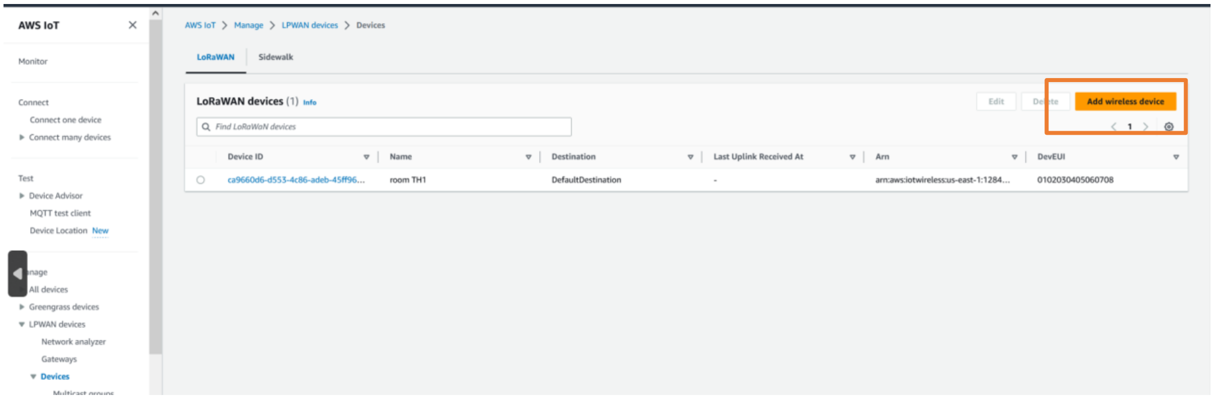
Devices
LoRaWAN devices connect to customers' gateways via LoRa communication protocol.

Gateways
Gateways connect to AWS IoT Core using LoRa Basic Station protocol over Secure WebSockets

AWS IoT Core for LoRaWAN
Securely and easily connect LoRaWAN devices to the cloud

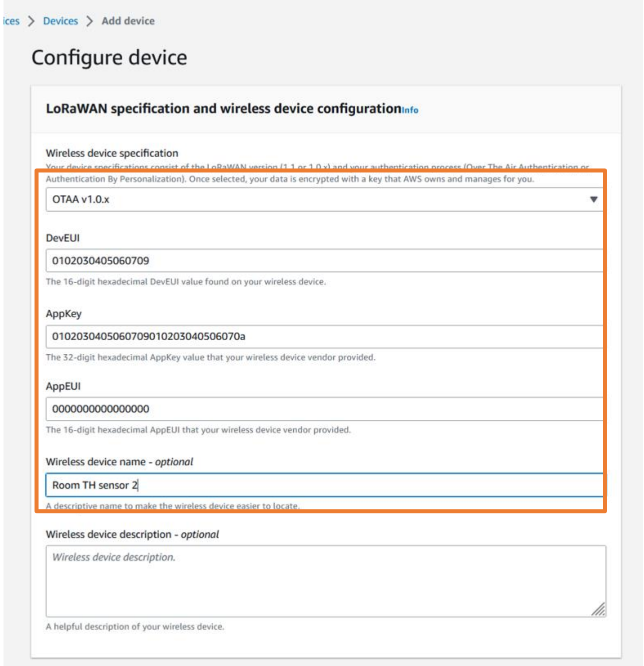
AWS Cloud Services
Messages are routed via AWS IoT Core Rules Engine to other AWS services

4.1.2 Add a wireless device



	Device ID	Name	Destination	Last Uplink Received At	Arn	DevEUI
○	ca9660d6-d553-4c86-adeb-45ff96...	room TH1	DefaultDestination	-	arn:aws:iotwireless:us-east-1:1284...	0102030405060708

4.1.3 Enter the DevEui, AppKey and the AppEui



ices > Devices > Add device

Configure device

LoRaWAN specification and wireless device configuration Info

Wireless device specification

Your device specifications consist of the LoRaWAN version (1.1 or 1.0.x) and your authentication process (Over The Air Authentication or Authentication By Personalization). Once selected, your data is encrypted with a key that AWS owns and manages for you.

OTAA v1.0.x

DevEUI

0102030405060709

The 16-digit hexadecimal DevEUI value found on your wireless device.

AppKey

0102030405060709010203040506070a

The 32-digit hexadecimal AppKey value that your wireless device vendor provided.

AppEUI

0000000000000000

The 16-digit hexadecimal AppEUI that your wireless device vendor provided.

Wireless device name - *optional*

Room TH sensor 2

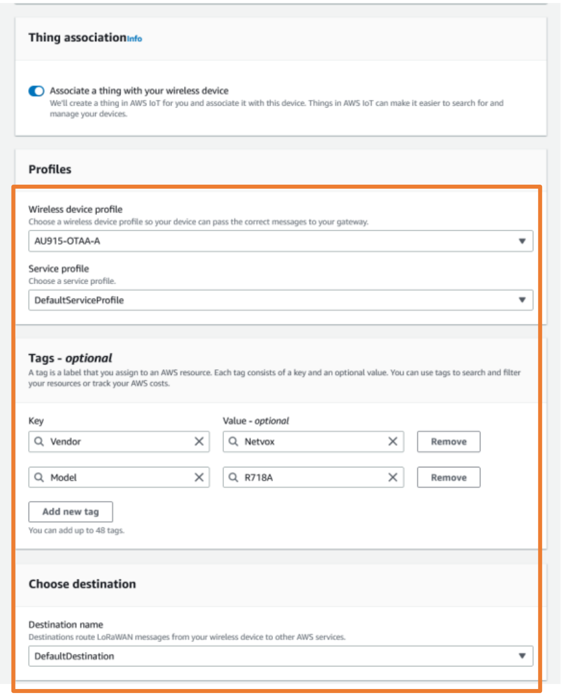
A descriptive name to make the wireless device easier to locate.

Wireless device description - *optional*

Wireless device description.

A helpful description of your wireless device.

4.1.4 Set device profile, service profile and destination



Thing association Info

Associate a thing with your wireless device

We'll create a thing in AWS IoT for you and associate it with this device. Things in AWS IoT can make it easier to search for and manage your devices.

Profiles

Wireless device profile

Choose a wireless device profile so your device can pass the correct messages to your gateway.

AU915-OTAA-A

Service profile

Choose a service profile.

DefaultServiceProfile

Tags - *optional*

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value - *optional*	
Vendor	Netvox	Remove
Model	R718A	Remove

You can add up to 48 tags.

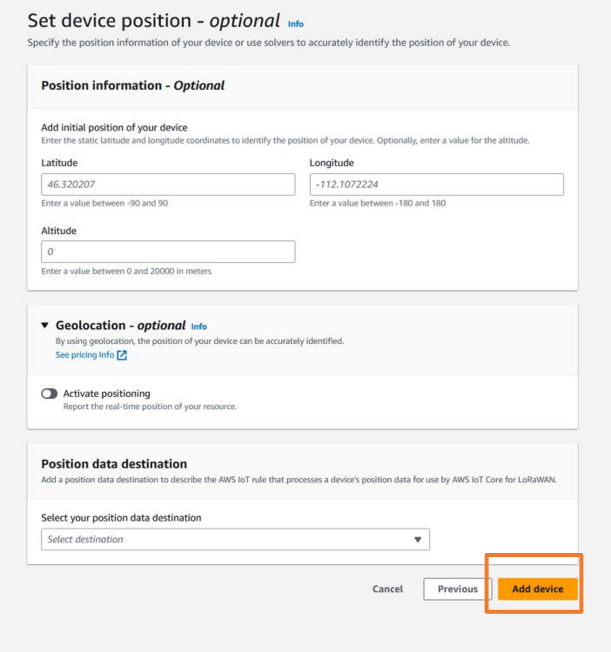
Choose destination

Destination name

Destinations route LoRaWAN messages from your wireless device to other AWS services.

DefaultDestination

4.1.5 Skip device positioning



Set device position - *optional* Info

Specify the position information of your device or use solvers to accurately identify the position of your device.

Position information - *Optional*

Add initial position of your device

Enter the static latitude and longitude coordinates to identify the position of your device. Optionally, enter a value for the altitude.

Latitude

46.320207

Enter a value between -90 and 90

Longitude

-112.1072224

Enter a value between -180 and 180

Altitude

0

Enter a value between 0 and 20000 in meters

Geolocation - *optional* Info

By using geolocation, the position of your device can be accurately identified.

See pricing Info

Activate positioning

Report the real-time position of your resource.

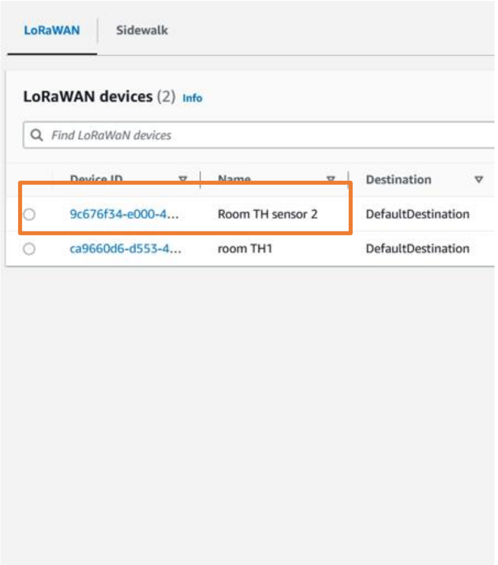
Position data destination

Add a position data destination to describe the AWS IoT rule that processes a device's position data for use by AWS IoT Core for LoRaWAN.

Select your position data destination

Select destination

4.1.6 Device Created



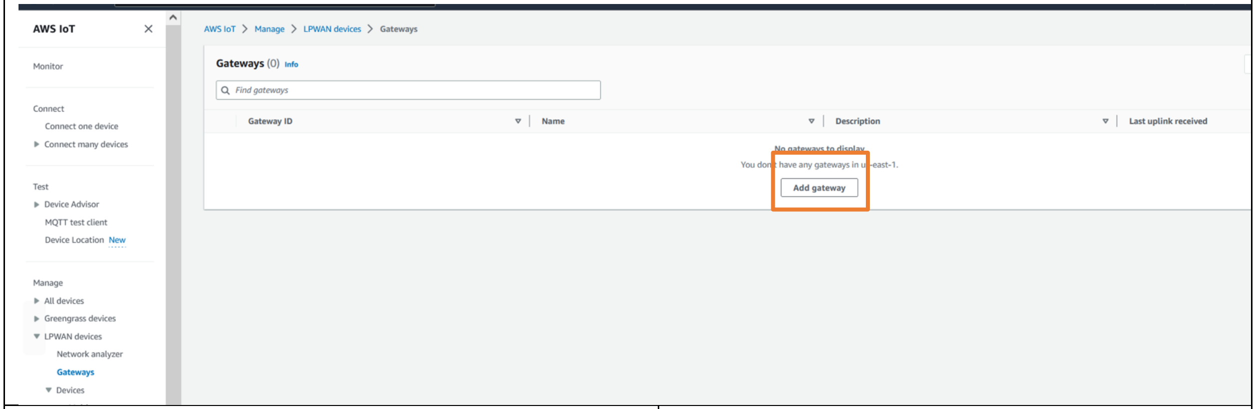
LoRaWAN devices (2) Info

Find LoRaWaN devices

Device ID	Name	Destination
9c676f34-e000-4...	Room TH sensor 2	DefaultDestination
ca9660d6-d553-4...	room TH1	DefaultDestination

4.2 Provision a LoRaWAN gateway

4.2.1 Navigate to Gateways



AWS IoT

Monitor

Connect
- Connect one device
- ▸ Connect many devices

Test
- ▸ Device Advisor
- MQTT test client
- Device Location New

Manage
- ▸ All devices
- ▸ Greengrass devices
- ▾ LPWAN devices
  - Network analyzer
  - Gateways
  - ▾ Devices

AWS IoT > Manage > LPWAN devices > Gateways

Gateways (0) Info

Find gateways

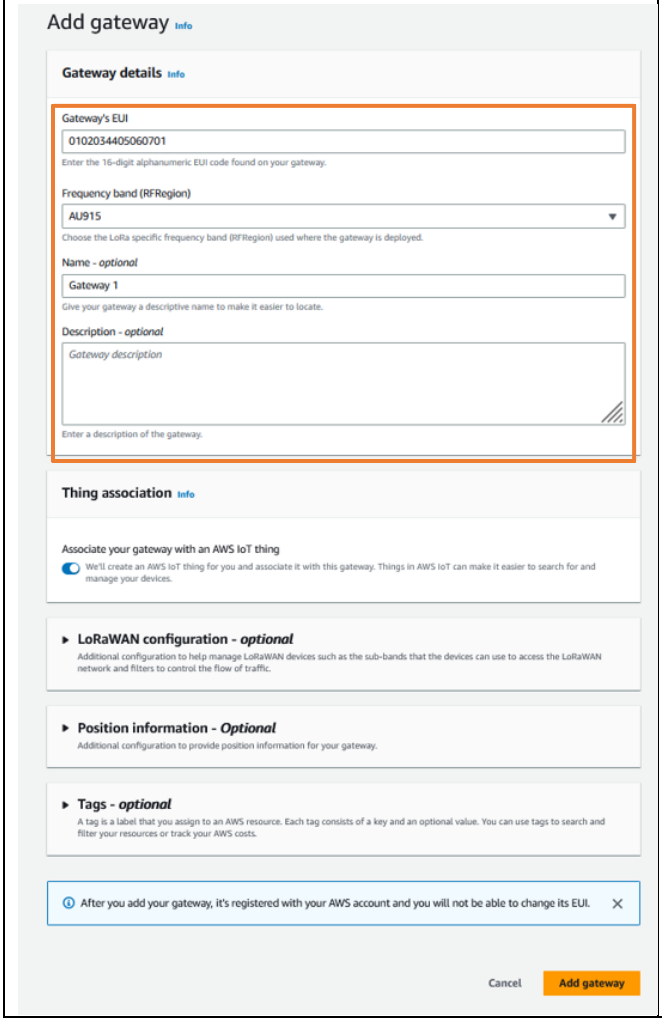
Gateway ID	Name	Description	Last uplink received

No gateways to display

You don't have any gateways in us-east-1.

Add gateway

4.2.2 Add Gateway ID and frequency band



Add gateway Info

Gateway details Info

Gateway's EUI

0102034405060701

Enter the 16-digit alphanumeric EUI code found on your gateway.

Frequency band (RFRegion)

AU915 ▾

Choose the LoRa specific frequency band (RFRegion) used where the gateway is deployed.

Name - *optional*

Gateway 1

Give your gateway a descriptive name to make it easier to locate.

Description - *optional*

Gateway description

Enter a description of the gateway.

Thing association Info

Associate your gateway with an AWS IoT thing

● We'll create an AWS IoT thing for you and associate it with this gateway. Things in AWS IoT can make it easier to search for and manage your devices.

▸ **LoRaWAN configuration - *optional***

Additional configuration to help manage LoRaWAN devices such as the sub-bands that the devices can use to access the LoRaWAN network and filters to control the flow of traffic.

▸ **Position information - *Optional***

Additional configuration to provide position information for your gateway.

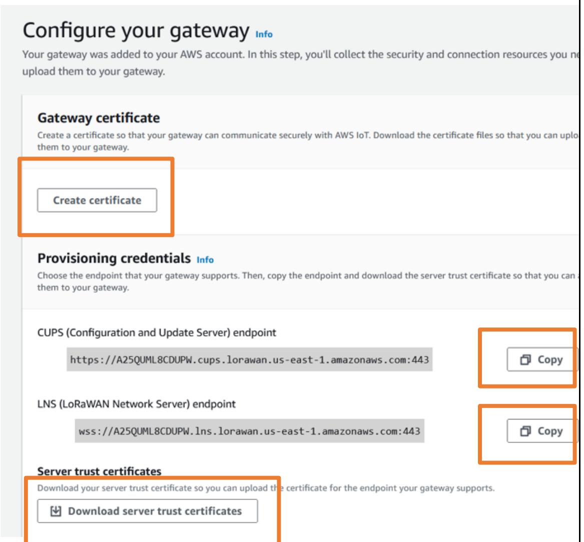
▸ **Tags - *optional***

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

ⓘ After you add your gateway, it's registered with your AWS account and you will not be able to change its EUI. ✕

Cancel Add gateway

4.2.3 Create certificate, save the CUPS & LNS URLs and download server trust certificates



Configure your gateway Info

Your gateway was added to your AWS account. In this step, you'll collect the security and connection resources you ne upload them to your gateway.

Gateway certificate

Create a certificate so that your gateway can communicate securely with AWS IoT. Download the certificate files so that you can uplo them to your gateway.

Create certificate

Provisioning credentials Info

Choose the endpoint that your gateway supports. Then, copy the endpoint and download the server trust certificate so that you can them to your gateway.

CUPS (Configuration and Update Server) endpoint

```
https://A25QUML8CDUPW.cups.lorawan.us-east-1.amazonaws.com:443
```

Copy

LNS (LoRaWAN Network Server) endpoint

```
wss://A25QUML8CDUPW.lns.lorawan.us-east-1.amazonaws.com:443
```

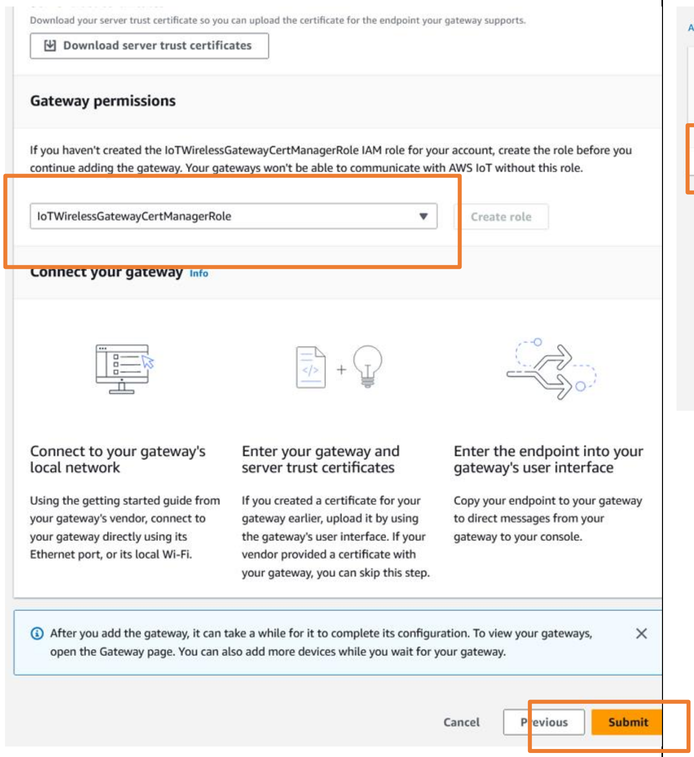
Copy

Server trust certificates

Download your server trust certificate so you can upload the certificate for the endpoint your gateway supports.

Download server trust certificates

4.2.4 Select the Certificate Manager Role and submit



Download your server trust certificate so you can upload the certificate for the endpoint your gateway supports.

Gateway permissions

If you haven't created the IoTWirelessGatewayCertManagerRole IAM role for your account, create the role before you continue adding the gateway. Your gateways won't be able to communicate with AWS IoT without this role.

IoTWirelessGatewayCertManagerRole ▾

Connect your gateway

Connect to your gateway's local network

Using the getting started guide from your gateway's vendor, connect to your gateway directly using its Ethernet port, or its local Wi-Fi.

Enter your gateway and server trust certificates

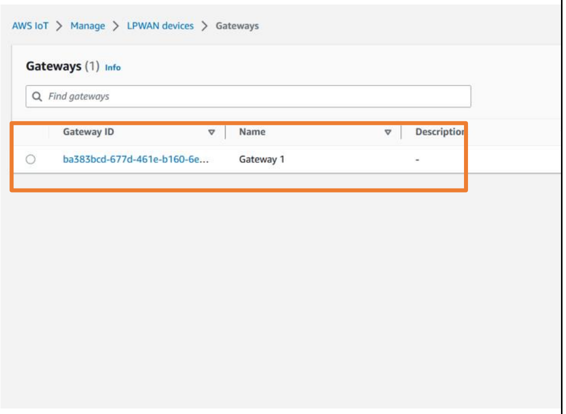
If you created a certificate for your gateway earlier, upload it by using the gateway's user interface. If your vendor provided a certificate with your gateway, you can skip this step.

Enter the endpoint into your gateway's user interface

Copy your endpoint to your gateway to direct messages from your gateway to your console.

After you add the gateway, it can take a while for it to complete its configuration. To view your gateways, open the Gateway page. You can also add more devices while you wait for your gateway.

4.2.5 Gateway is now created



	Gateway ID	Name	Descriptio
○	ba383bcd-677d-461e-b160-6e...	Gateway 1	-

4.3 Provision a MQTT sensor

The process to provision a MQTT sensor is the same standard process as creating a thing in AWS IoT Core. Refer the official AWS documentation here: https://docs.aws.amazon.com/iot/latest/developerguide/iot-moisture-setup.html

The sensor should publish to a topic of the form:

<sensor DevEui>/tx

The IoT Core's Generic Sensor Rule picks the dev Eui from the publish topic and writes the uplink JSON message's properties as separate measures to the sensor_data table in the Timestream database

5. Troubleshooting guide

5.1 Stack deployment failed

- Does your AWS account have the permissions to create the resources indicated in the architecture diagram?

5.2 Gateway provisioning failed

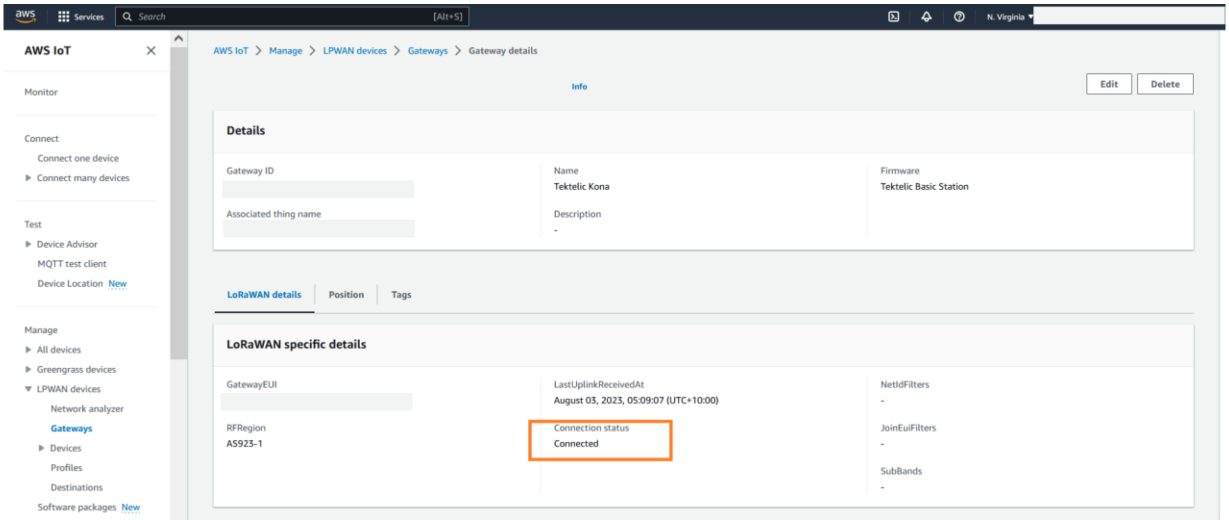
- Is this gateway provisioned already in a this or different AWS account?
- Was the correct IoT Wireless Certificate Manager role selected when provisioning the gateway?

5.3 Sensor provisioning failed

- Is a sensor already provisioned in this AWS account?

5.4 No data from sensors

- Is the sensor powered up?
- Is the gateway connected to IoT Core? This can be checked from the gateway page in IoT as shown below.



Details

Gateway ID

Associated thing name

Name
Tektelic Kona

Description
-

Firmware
Tektelic Basic Station

LoRaWAN specific details

GatewayEUI

RFRegion
AS923-1

LastUplinkReceivedAt
August 03, 2023, 05:09:07 (UTC+10:00)

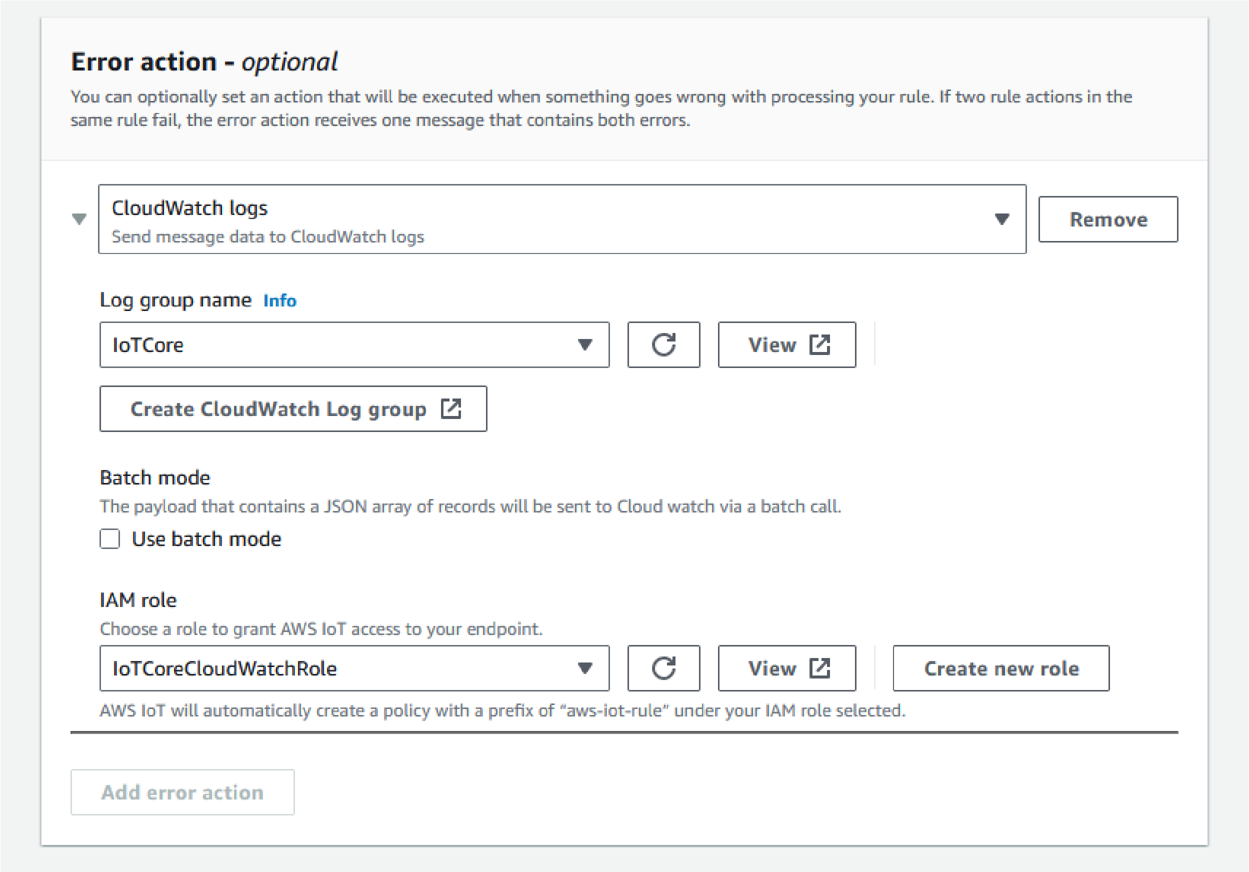
Connection status
Connected

NetIdFilters
-

JoinEuiFilters
-

SubBands
-

- Is the sensor provisioned correctly with the dev eui, app eui, app key and destination?
- Does the sensor have the ‘Vendor’ and ‘Model’ tags?
- Is the vendor and model supported in this accelerator? If not, the decoder function will need to be created in the decoder Lambda function
- Create a log group and enable Error logs in IoT Core for the ‘DefaultIoTRule_xxxxxxxxxxxx” as shown below:



Error action - ***optional***

You can optionally set an action that will be executed when something goes wrong with processing your rule. If two rule actions in the same rule fail, the error action receives one message that contains both errors.

CloudWatch logs
Send message data to CloudWatch logs

Log group name: IoTCore

Batch mode
The payload that contains a JSON array of records will be sent to Cloud watch via a batch call.

□ Use batch mode

IAM role
Choose a role to grant AWS IoT access to your endpoint.

IoTCoreCloudWatchRole

AWS IoT will automatically create a policy with a prefix of "aws-iot-rule" under your IAM role selected.

If there are error logs, it means everything up to this point in the uplink path is fine. Once the error is resolved, move to the next step

- Navigate to CloudWatch and check the logs for the 'iotsensordash-DefaultDecoderLambda-xxxxxxxxxxxx'.
  - Do they indicate any errors during decoding or writing to Timestream?
  - If there aren't any logs for the Lambda it means that the Lambda is not getting invoked. This implies that the IoT Core rule is not getting invoked.
- Navigate to Timestream and run a query on the sensor_data table for the DevEui of the sensor.

Eg: select * from 'iotsensordash'.'sensor_data' where DevEui='xxxxxxxxxxxxxxxx'

This should bring up the data received and stored in Timestream.

5.5 Sensor not listed in the template dashboard dropdown

- Was the sensor created with the 'Vendor' and 'Model' tags? If not, delete the sensor and re-create the sensor. Adding the tags after creation will **NOT** work.

6. AWS Cost estimate

A cost estimate can be found here:

https://calculator.aws/#/estimate?id=c04beb91e36fcc41acbc64c97dce9ae09263d945

Assumptions:

- Region: ap-southeast-2 (Sydney)
- 100 LoRaWAN devices
- 1 message per device every 10mins
- 1 dashboard live 24x7, auto refreshed with 4 queries to Timestream every 10mins
- 10 alarm queries run every 10mins
- 1 Grafana Admin user and 1 Viewer

Cost summary:

Service	Monthly Cost (USD)
AWS IoT Core	1.35
AWS Lambda	0.99
Amazon Timestream	9.01
Amazon Managed Grafana	14.00