Quickstart - Everynet to Azure IoT Hub

Quickly integrate Everynet connected devices to Azure IoT Hub

Everynet to Azure IoT Hub

This guide will walk you though integrating a LORA device connected to the Everynet LNS to Azure IoT Hub using solution templates. This guide involves importing multiple solution templates depending on which LORA device type you are looking to integrate.

Before starting this guide, be sure to read the deep dive here: LORA-to-AzureIoT

Pre-requisites

1. Account on Tartabit IoT Bridge.
2. Access to a Microsoft Azure subscription.
3. LORAWAN device communicating to an Everynet LNS.
4. Information to create a decoder for your sensor.

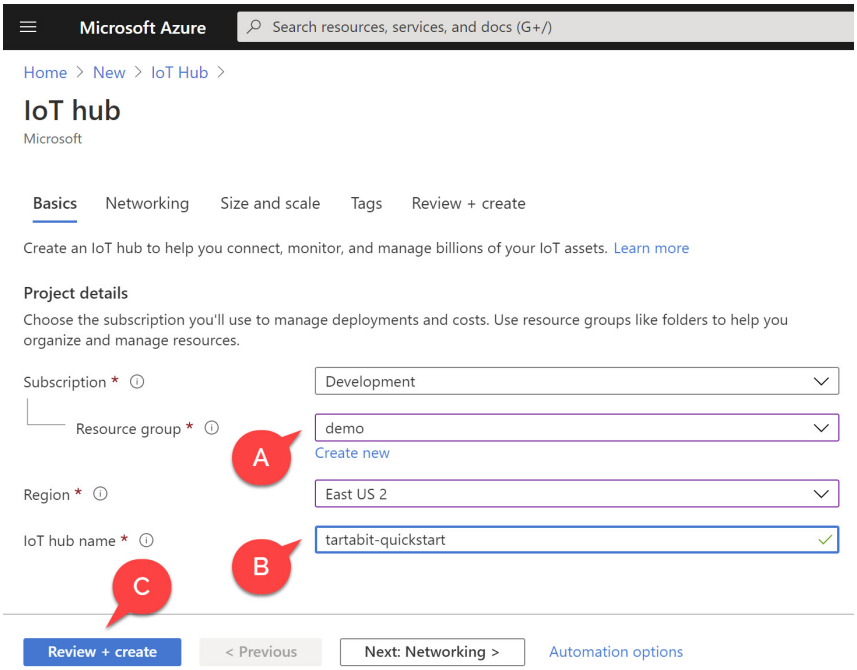
Details of this guide

- In this guide you will:
  - Create a new Azure IoT Hub.
  - Retrieve the credentials needed to connect to your Azure IoT Hub.
  - Import the basic solution template, this will create the Azure IoT service connector, and the basic triggers for processing LORA data.
  - Connect an Everynet LNS to the IoT Bridge
  - Connect a LORAWAN device
  - Verify that the device is properly reporting into the Azure IoT Hub.

1. Create an IoT Hub in Azure

If you already have an IoT Hub, you can skip this step.

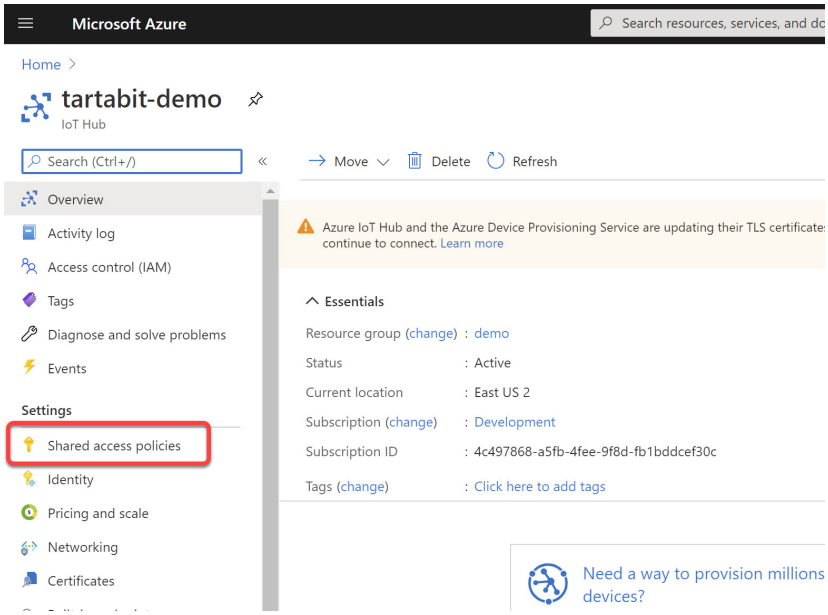
For a detailed walkthrough from Microsoft, check out the following link:
https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-create-through-portal



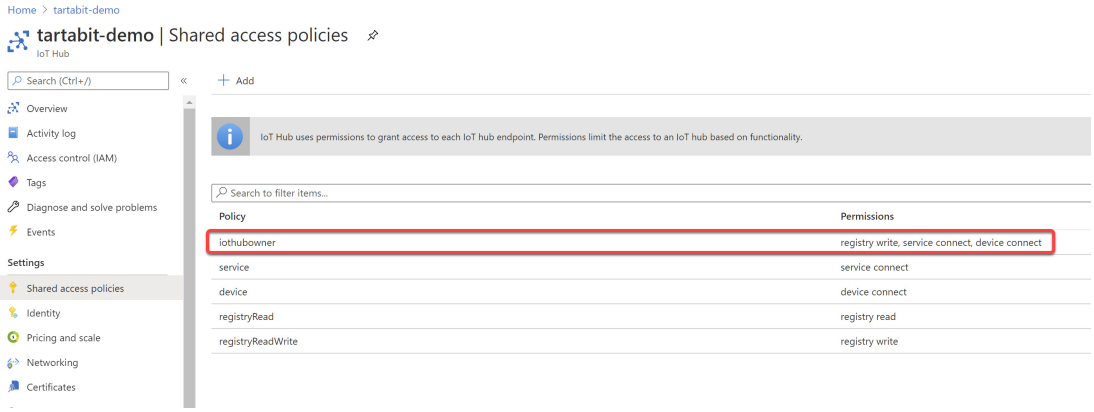
- A: Select an existing resource group or create one.
- B: Provide a unique name for your IoT Hub.
- C: You can skip the other tabs and create a new Hub.

2. Retrieve your IoT Hub connection string

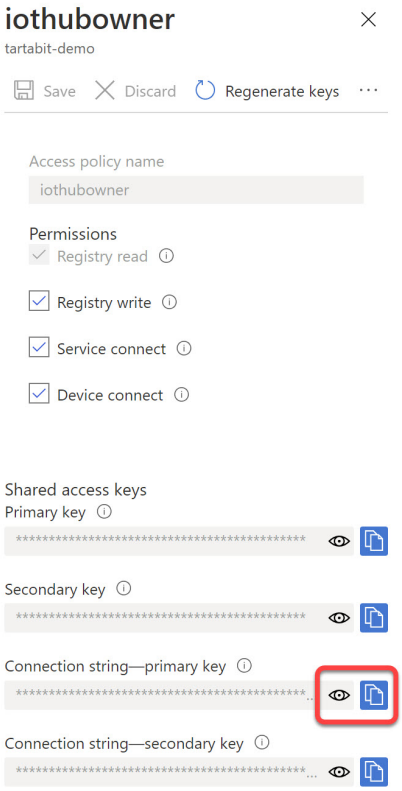
1. Navigate to your new Azure IoT Hub.
2. Click **Shared Access Policies**.



3. Select **iothubowner** (this is required because the IoT Bridge will automatically provision devices in the IoT Hub as they are needed).



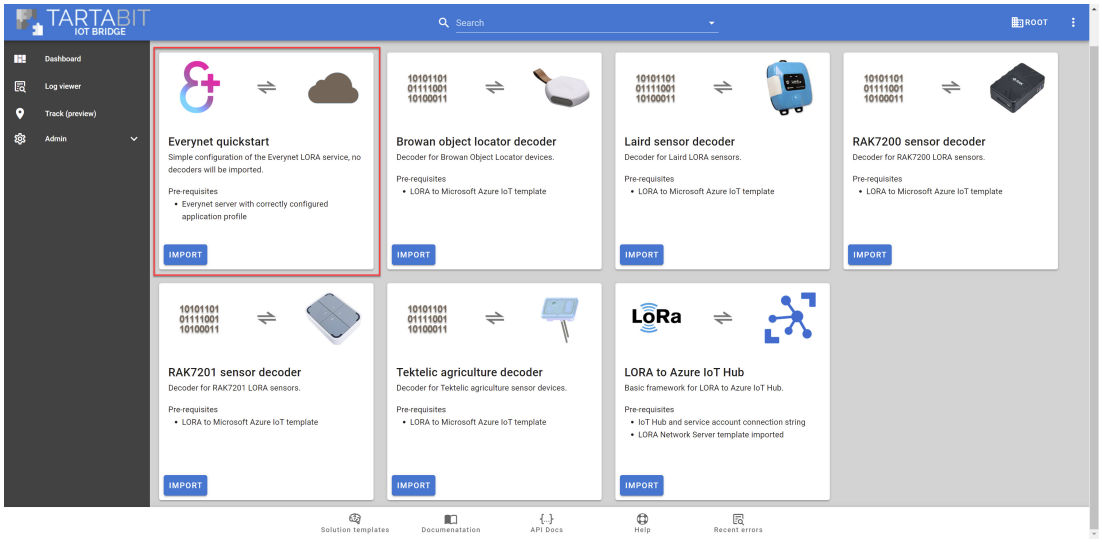
4. Copy the **Connection string - primary key** value and save it for later.



3. Configure the Everynet Server

Import the solution template

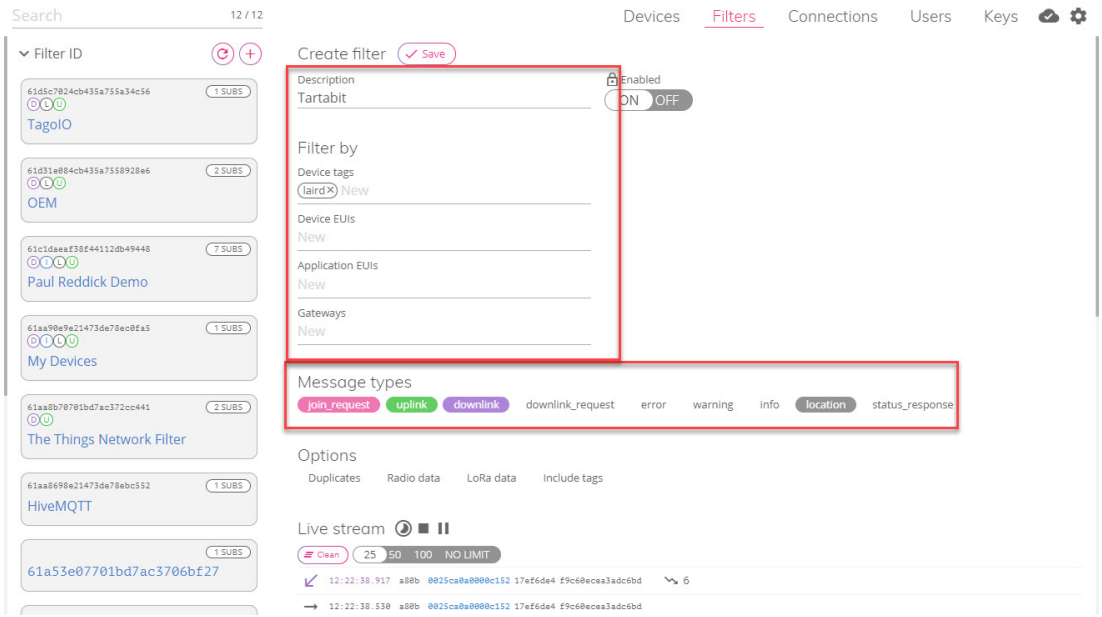
1. Import the `Everynet quickstart` solution template by clicking the Import button.



2. Make note of the webhook secret, you will need it when configuring the Everynet HTTP Connection.

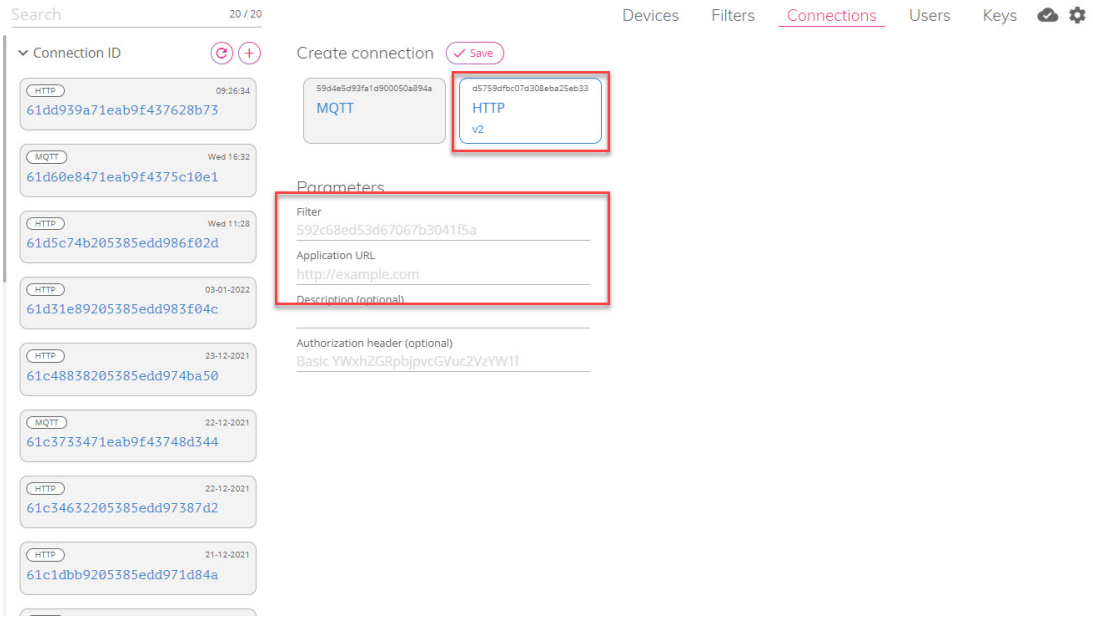
Configure Everynet LNS

1. Login to your Everynet server.
2. Navigate to `Filters` on the main menu.



3. Configure a new filter for the devices you want to send to the IoT Bridge. Supported message types are `join_request`, `uplink`, `downlink`, and `location`.
4. Save your filter and store the ID assigned to the new filter.

5. Navigate to `Connections` on the main menu.



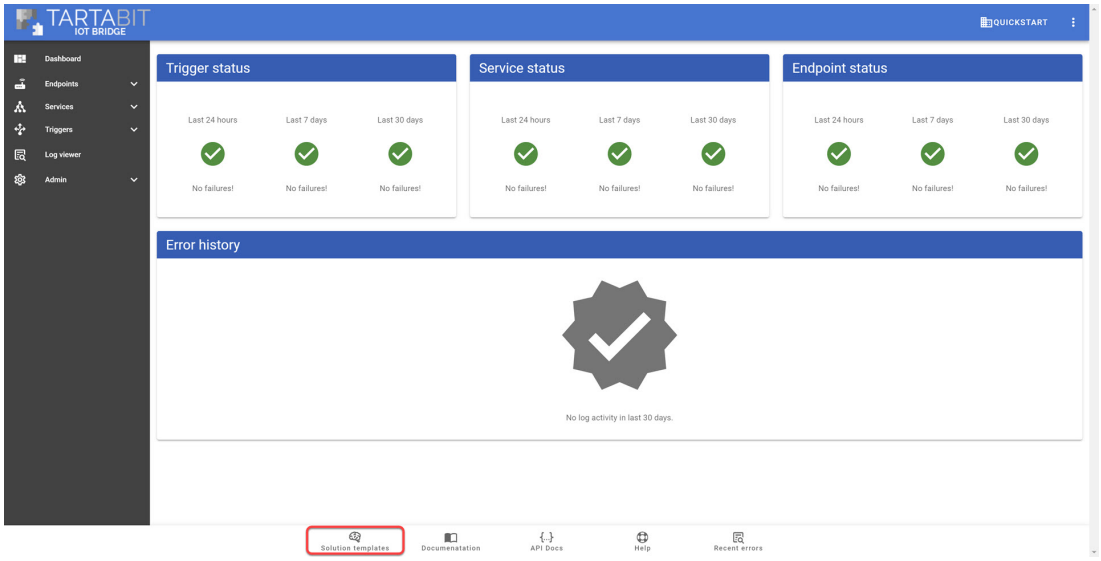
6. Create a new connection, select the type as `HTTP`, set the filter to the ID you stored from step 4, and set the application URL to https://bridge-us.tartabit.com/webhook/ <webhook from above>

After configuring the LORA Network Server, continue to the next step.

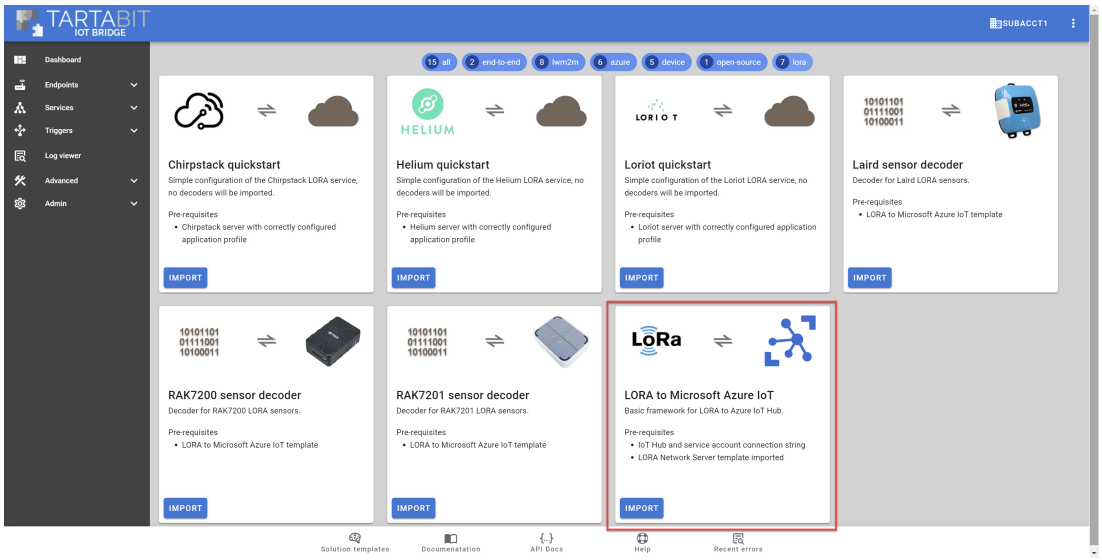
4. Import the basic solution template

Follow the steps below to import the solution template:

1. Click **Solution Templates** at the bottom of any page.



2. Find the **LORA to Microsoft Azure IoT Hub** template and click **Import**.



3. Follow the instructions below to import the template:

- Enter the Azure IoT Hub connection string that you previously saved in Step #1.
- Select the option to **Start Triggers** this will automatically start all of the new triggers upon import.
- Click import when finished.

5. Configure decoders for your LORA sensors

There are several ways to configure decoders for your LORA sensors, you can select from the following:

Use a pre-configured decoder solution template

There are several templates already available for common devices, and new ones are being added as needed. Check out the solution templates and filter on `LORA` and `Device` and import these templates.

- The pre-configured decoders are written to work with the `LORA to Azure IoT Hub` template, after importing them you do not need to modify the triggers.
- You MUST modify the `Decode Sensor Data` trigger to add logic to route to the decoder. You can see a sample commented out that shows how to route based on ports.

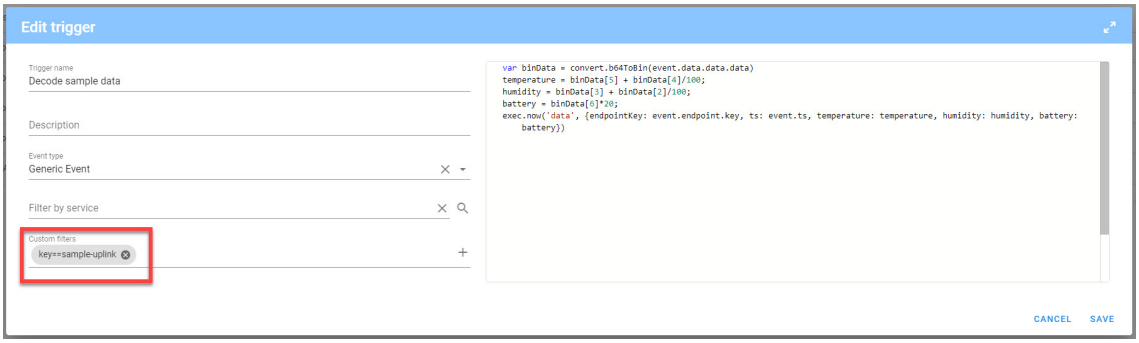
Write a custom decoder trigger

The trigger scripts use a javascript runtime, you can decode the payload in the script. Check out the `Decode Sample Data` trigger for an example of where to start. Here are the key considerations when creating a custom decoder:

- Your goal is to decode the binary data (that comes in as a base64 encoded string) and output a javascript object that can be further processed.
- You can generally copy/paste decoders written for TTN into the triggers, this can accelerate development by re-using pre-existing scripts.

Each decoder must be given a unique filter and added to the `Decode Sensor Data` trigger

Each decoder trigger must be given a unique filter based on the key for the generic event. This is to ensure that you can route events from the `LORA Message Router` trigger to the correct decoder. Additionally, you must add logic to the `LORA Message Router` trigger to route traffic to your decoder. The most common way to manage multiple different sensor types is to ensure they are publishing on separate data ports.



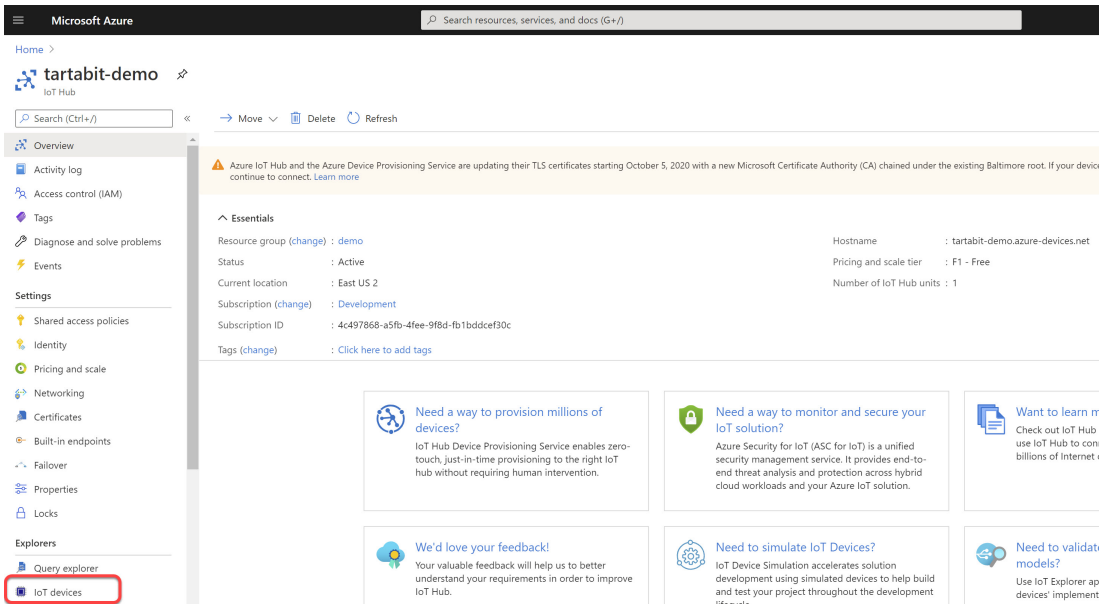
6. Start sending data

You now should be able to start sending data from your devices and see that data work its way through the system. As your devices transmit, you will see activity registered in the IoT Bridge.

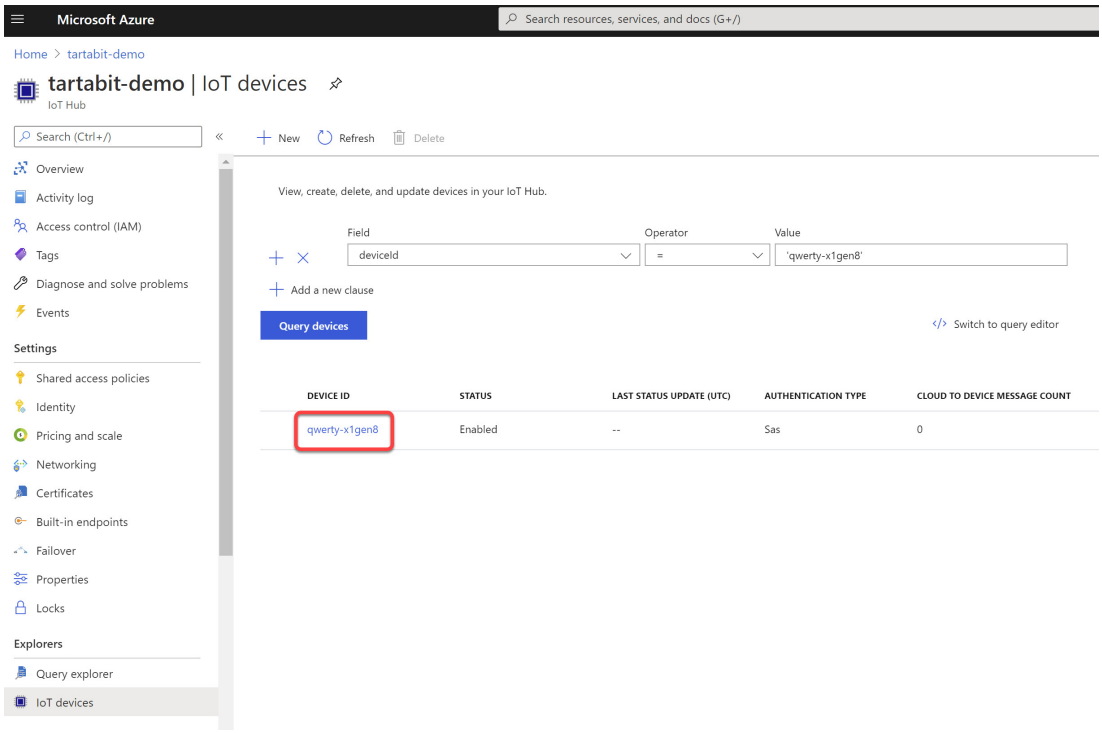
7. Check your device in Azure IoT Hub

With the device connected, you should now see it connected in the IoT Hub, and data being updated in the device twin.

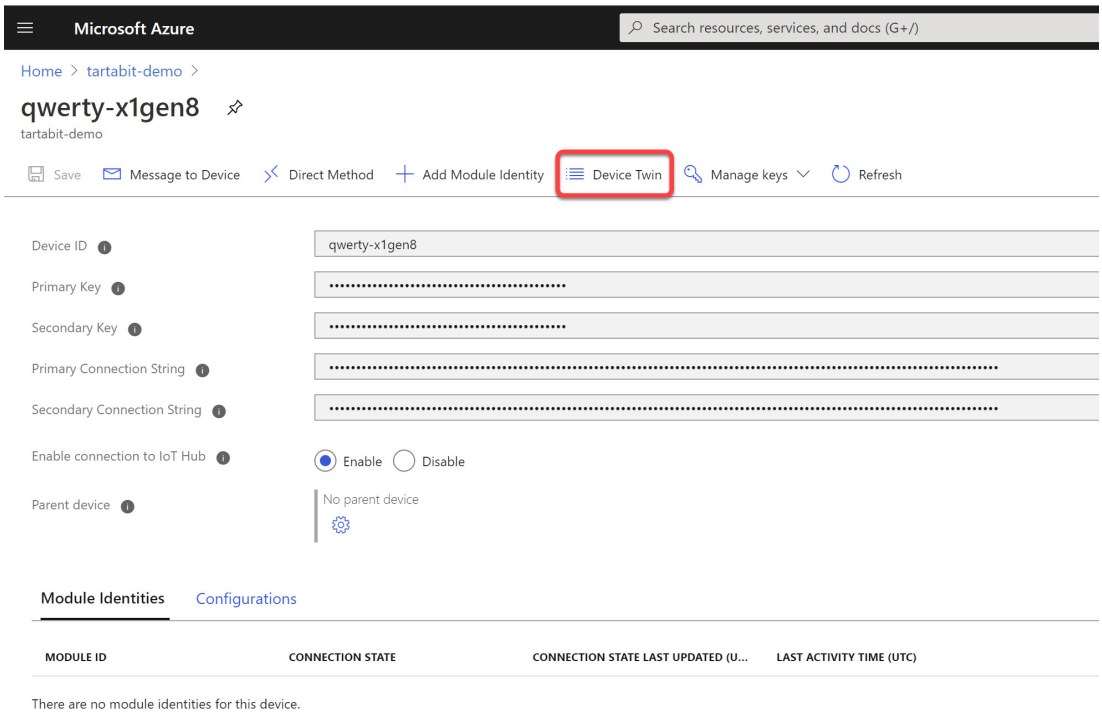
1. Navigate to your Azure IoT Hub
2. Click **IoT Devices**



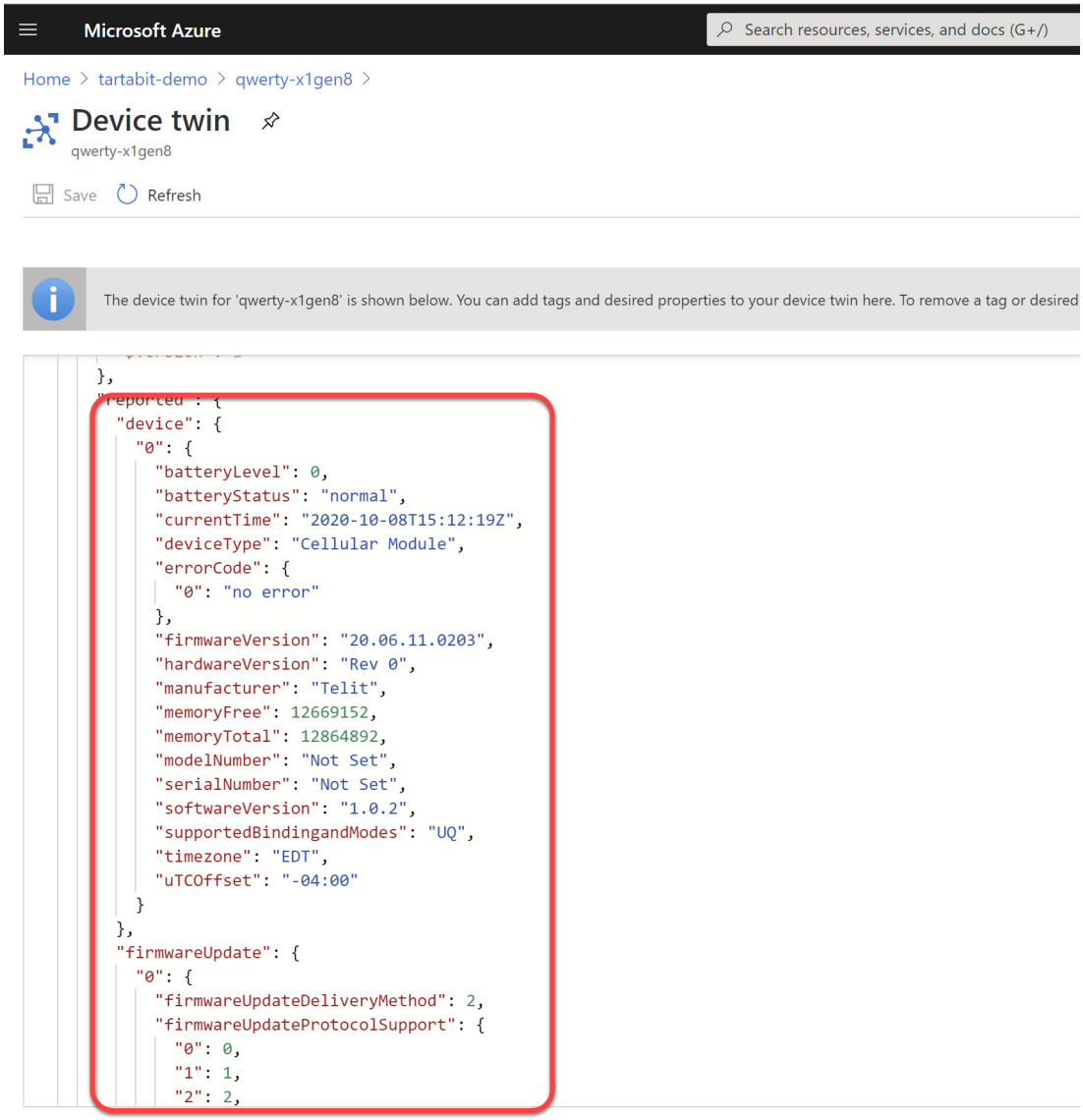
3. You should see your newly created device, click on it.



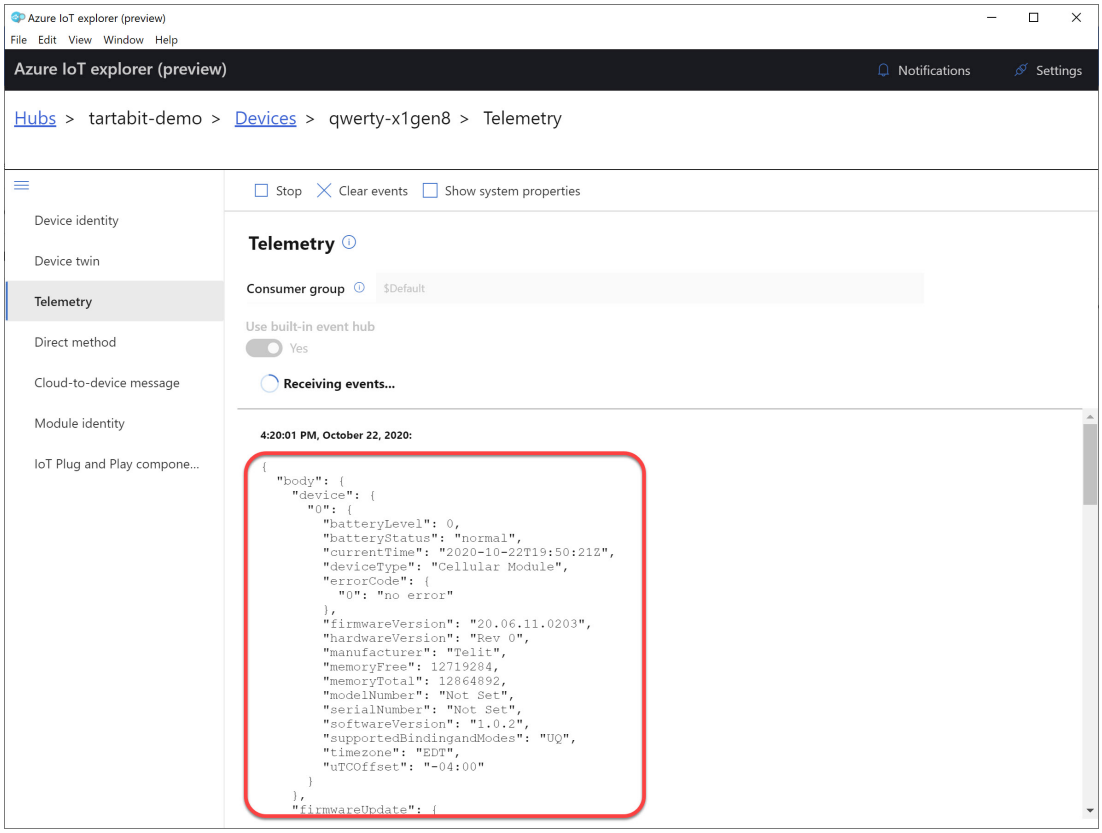
4. Click on **Device Twin**



5. You will see the sensor in your twin.



6. You can use the Azure IoT Explorer to view the telemetry as well.



You are done!