Connect Raspberry Pi to Azure IoT Hub (Node.js)

Jim Bennett, Microsoft Developer Advocate

Updated: 2 months ago | Contributors: 4

In this lesson, you begin by learning the basics of setting up the Raspberry Pi that's running Raspbian. You then learn how to seamlessly connect your device to the cloud by using Azure IoT Hub. Finally, you'll use Azure IoT to write the Internet-of-Things (IoT) solution.

Don't have a IoT kit? Try Raspberry Pi with simulators. Or buy a kit like here.

What you do

- Set up your device to run Azure IoT Hub and get your new device connection string.
- Write a simple IoT app in JAVASCRIPT.
- Run a sample application on Pi and remote-control device to your IoT hub.

What you learn

- How to connect an Azure IoT Hub and get your new device connection string.
- How to connect IoT with a JAVASCRIPT app.
- How to collect sensor data by running a sample application on Pi.
- How to remote-control device to your IoT hub.

What you need

- Raspberry Pi 3 or Raspberry Pi 4 board.
- An Azure subscription. If you don't have an Azure subscription, create a free account before you begin.
- A monitor, a USB keyboard, and mouse that connects to Pi.
- A 16-GB or larger microSD card to hold the Raspbian OS.
- An internet connection.
- A 5-V, 2.5-A power supply with USB-C connector.
- A USB-C power cable (microSD card to boot the operating system image onto the microSD card).
- A 5-V, 2.5-A power supply with micro-B USB-C to micro-USB cable.

Optional

If you don't have the optional items, you can use simulated device data.

- A Raspberry Pi 3 or Raspberry Pi 4 board.
- An Azure subscription. If you don't have an Azure subscription, create a free account before you begin.
- A monitor, a USB keyboard, and mouse that connects to Pi.
- A 16-GB or larger microSD card to hold the Raspbian OS.
- An internet connection.
- A 5-V, 2.5-A power supply with USB-C connector.
- A USB-C power cable (microSD card to boot the operating system image onto the microSD card).
- A 5-V, 2.5-A power supply with micro-B USB-C to micro-USB cable.

Optional

If you don't have the optional items, you can use simulated device data.

Create an IoT hub

1. Sign in to the Azure portal.
2. Select **Create a resource** > **Internet of Things** > **IoT Hub**.

3. In the **IoT Hub** pane, enter the following information for your IoT hub:
   - **Subscription**: Choose the subscription that you want to use to create this IoT hub.
   - **Resource group**: Create a resource group to host the IoT hub or use an existing one. For more information, see Use resource groups to manage your Azure resources.
   - **Region**: Select a closest location to you.
   - **IoT hub name**: Enter a name for your IoT hub. This name must be unique, a green check mark appears.

   Important

   The IoT hub will be publicly discoverable as a DNS endpoint, so make sure to avoid any sensitive information while naming it.

4. Select **Next: Size and scale** to continue creating your IoT hub.
5. For **Pricing and scale tier**, select your tier. For this tutorial, select **F1: Free tier** if it's still available on your subscription. For more information, see the Pricing and tiers.

   Note

   You can select the subscription connection settings for free per an hourly1. However, you may reach the limit of the number of allowed IoT connects after you complete this set up.

6. Select **Review + create**.
7. Review your IoT hub information. Check that **Costs** from IoT hub might incur a fee minutes to create. You can monitor the charges in the **Notifications** pane.

Now that you have created an IoT hub, locate the important information that you use to connect your device and your application to IoT hub.

In your IoT hub navigation menu, open **Shared access policies**. Select the **iothubowner** policy, and then copy the **Connection string—primary key** of your IoT hub. For more information, see Control access to IoT hubs.

   Note

   Keep a record the iothubowner connection string for this set up tutorial. However, you may need to the value of the iothubowner IoT connects after you complete this set up.

Register your device in the IoT hub

1. In your IoT hub navigation menu, open **IoT devices**, then click **Add** to register a device to your IoT hub.

2. Enter a **Device ID** for the new device. Device IDs are case sensitive.

   Important

   The device ID may be visible in the logs collected for customer support and troubleshooting, so make sure to avoid any sensitive information while naming it.

3. Click **Save**.
4. After the device is created, open the device from the list in the **IoT devices** pane.
5. Copy the **Connection string—primary key** to use later.

Set up Raspberry Pi

Install the Raspbian operating system for Pi

Prepare the microSD card for installation of the Raspbian image:

1. Download Raspbian.
   - Download Raspbian Desktop (Zip) zip file.
   - Warning

     Make sure you download the download. **raspbian.zip** or **.zip** image. The latest version of Raspbian images have moved from Stretch to Buster and Bullseye. Buster and Bullseye image versions in your next steps.

2. Install the Raspbian image to a folder on your computer.
   - Install Raspbian to the microSD card.
   - Download and install the Raspbian image that you prepared in step 1.
   - Select the microSD card drive. Enter your keys already stored in the current drive.
   - Click Next to install Raspbian to the microSD card.
   - Remove the microSD card from your computer when installation is complete. It's safe to remove the microSD card directly because Driver automatically ejects or unmounts the microSD card when installation.
   - Insert the microSD card into Pi.

Enable SSH and I2C

1. Connect Pi to the monitor, keyboard, and mouse.
2. Power on Pi. Wait for Raspbian to boot up, or, on first-time setup and **raspbian**, or the pre-installed.
3. Click the Raspberry icon > **Preferences** > **Raspberry Pi Configuration**.

4. On the **Interfaces** tab, set **SSH** and **I2C** to **Enable**, and then click **OK**. If you don't have physical sensors and want to use simulated sensor data, skip step 5 and 6.

   Note

   To enable SSH and I2C, you can find more reference documents on raspberry-pi.org and **raspbian.com**.

Connect the sensor to Pi

Use the breadboard and jumper wires to connect an LED and a BME280 to Pi as follows. If you don't have the sensor, skip to the next section.

Pin	Pin	Pin
GPIO Pin 17	Pin 1	Pin 1
GPIO Pin 2	Pin 2	Pin 2
GPIO Pin 3	Pin 3	Pin 3
GPIO Pin 4	Pin 4	Pin 4
GPIO Pin 5	Pin 5	Pin 5
GPIO Pin 6	Pin 6	Pin 6
GND Pin 7	Pin 7	Pin 7
GND Pin 8	Pin 8	Pin 8
GND Pin 9	Pin 9	Pin 9
GND Pin 10	Pin 10	Pin 10
GND Pin 11	Pin 11	Pin 11
GND Pin 12	Pin 12	Pin 12
GND Pin 13	Pin 13	Pin 13
GND Pin 14	Pin 14	Pin 14
GND Pin 15	Pin 15	Pin 15
GND Pin 16	Pin 16	Pin 16
GND Pin 17	Pin 17	Pin 17
GND Pin 18	Pin 18	Pin 18
GND Pin 19	Pin 19	Pin 19
GND Pin 20	Pin 20	Pin 20

Click to view Raspberry Pi 3 & 4 pin mapping for your reference.

After you've successfully connect the BME280 to your Raspberry Pi, it should look like below image.

Connect Pi to the network

Turn on Pi using the micro-USB cable and the power supply, use the Ethernet cable to connect to the network, or use the wireless network connection. If you don't have a wireless network, or if you don't have an Ethernet cable, skip to the next section.

   Note

   Make sure that Pi is connected to the same network as your computer. For example, if your computer is connected to a wireless network, then Pi should be connected to a wireless network, or if your computer is connected to a wired network, then Pi should be connected to a wired network.

Run a sample application on Pi

Clone sample application and install the prerequisite packages

1. Connect to your Raspberry Pi with one of the following SSH clients from your host computer:

   Windows clients

   - Download and install PuTTY for Windows.
   - After installation, open PuTTY. Enter the Pi host name or IP address and select port SSH as the connection type.

2. Install Node.js and NPM to your Pi.

```
Type: Enter
```

```
node -v
```

   If the version is lower than v.x, or if there is no Node.js on your Pi, install the latest version.

```
curl -sL https://deb.nodesource.com/setup_12.x | sudo -E bash
sudo apt-get -y install nodejs
```

3. Clone the sample application.

```
Type: Enter
```

```
git clone https://github.com/azure-iot/iot-hub-node-sample-client
```

4. Install all packages for the sample. The installation includes Azure IoT device SDK, BME280 Sensor library, and Wiring Pi library.

```
Type: Enter
```

```
cd iot-hub-node-sample-client; npm
sudo apt-get install
```

   Note

   It might take several minutes to finish this installation process depending on your network connection.

Configure the sample application

1. Open the config file by running the following commands.

```
Type: Enter
```

```
nano config.js\n
```


   There are two items in the file you can configure: The first one is **deviceID**, which defines the device identity in Azure IoT Hub. The second one is **deviceConnectionStringIoT**, which is a Boilerplate value for whether to use simulated sensor data or not.

   If you **deviceID** and **deviceConnectionStringIoT**, **deviceID**, **deviceConnectionStringIoT**, to make the sample application create and use simulated sensor data.

   Save and exit by typing Control-O > Enter > Control-X.

Run the sample application

Run the sample application by running the following command:

```
Type: Enter
```

```
node node index.js "[IoT HUB DEVICE CONNECTION STRING]"
```

   Note

   Make sure you copy paste the device connection string into the single quote.

You should see the following output that shows the sensor data and the messages that are sent to your IoT hub.

Next steps

You've run a sample application to collect sensor data and send it to your IoT hub. To see the messages that are received by your IoT hub, and to send messages from your IoT hub to your device, see Quickstart: Send telemetry from a device to IoT Hub and Quickstart: Send cloud-to-device messages with IoT Hub (Node.js).

To continue to get started with Azure IoT Hub and to explore other IoT scenarios, see the following:

- Discover cloud device management with Azure IoT Toolkit extension for Visual Studio Code
- Learn how Azure IoT Hub integrates its Device data sources
- Learn how to manage your device data with Azure Stream Analytics
- Learn how to connect your Azure Sphere devices to Azure IoT Hub using Azure Sphere Security Service
- Connect modified IoT control the sensor data from Azure IoT Central to Azure Machine Learning
- Monitor Windows with Azure IoT Suite's pre-configured Azure Studio Code
- Learn more, learn by sensors monitoring and notifications

Feedback

Was it helpful? Tell us about your experience.
Rate this page: Sign in to give documentation feedback
Tell us how we can improve this page.
Loading feedback...