

# Guide to deploying IoT-Sistem-Za-Kontrolu-Ulaza solution

Stokić Dušan

August 2021.

## Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Raspberry Pi device setup</b>	<b>3</b>
2.1	Installing Raspberry Pi OS . . . . .	3
2.2	Raspberry Pi quick start . . . . .	5
<b>3</b>	<b>Create Azure resources</b>	<b>6</b>
3.1	Create an IoT hub . . . . .	6
3.1.1	Register an IoT Edge device in IoT Hub . . . . .	7
3.2	Create a Storage Account for Table Storage . . . . .	9
3.3	Create a Stream Analytics job to Save Data in Table Storage . . . . .	9
3.3.1	Configure job input . . . . .	10
3.3.2	Configure job output . . . . .	10
3.3.3	Define the transformation query . . . . .	11
3.3.4	Create a Container Registry . . . . .	12

<b>4</b>	<b>IoT Edge on Raspberry Pi</b>	<b>13</b>
4.1	SSH into your Raspberry Pi . . . . .	13
4.2	Install IoT Edge runtime on Raspberry Pi . . . . .	14
4.3	Provision the device with its cloud identity . . . . .	15

# 1 Introduction

This is a detailed guide on deploying the solution in the IoT-Sistem-Za-Kontrolu-Ulaza repository<sup>1</sup>.

In this project you will learn how to use Microsoft Azure IoT Edge on a Raspberry Pi Model 3 B as an edge device with sensors attached to it. This project can be used as a base for larger Azure IoT Edge projects or as a Proof of Concept for Azure IoT Edge with real sensors.

This guide will cover setting up Azure services, installing the latest Raspian OS on the Raspberry Pi device and configuring the device as a IoT Edge device, connecting sensors and actuators and, finally, building and deploying the solution on your Edge device.

We are using a fresh installation of Raspberry Pi OS as the operating system but it might work with a not so fresh installation of Raspberry Pi. You can follow the Raspberry Pi OS installation guide in the next section.

## 2 Raspberry Pi device setup

This section will cover installing Raspberry pi OS (previous Raspbian) on your Raspberry Pi. After completing this section you will have a Raspberry Pi device up and running.

### 2.1 Installing Raspberry Pi OS

To get started with your Raspberry Pi computer you'll need A computer monitor, or television and a computer keyboard and mouse. Most monitors should work as a display for the Raspberry Pi, but for best results, you should use a display with HDMI input. You'll also need a appropriate display cable, to connect your monitor to your Raspberry Pi.

Finally you'll need an SD card. It is recommend to use a micro SD card with a minimum of 8GB, and to use an Imager to install an operating system onto it.

---

<sup>1</sup><https://github.com/StokicDusan/IoT-Sistem-Za-Kontrolu-Ulaza>

Raspberry Pi recommend the use of Raspberry Pi Imager<sup>2</sup> to install an operating system on your SD card. You will need another computer with an SD card reader to install the image. After installing the imager:

- Connect an SD card reader with the SD card inside.
- Open Raspberry Pi Imager and choose the recommended Raspberry Pi OS (32-bit) from the list presented.
- Choose the SD card you wish to write your image to.

After placing the SD card in your Raspberry Pi device, start it and begin the setup. Upon booting the device, you will be met with the welcome screen like in figure 1 and start the setup.

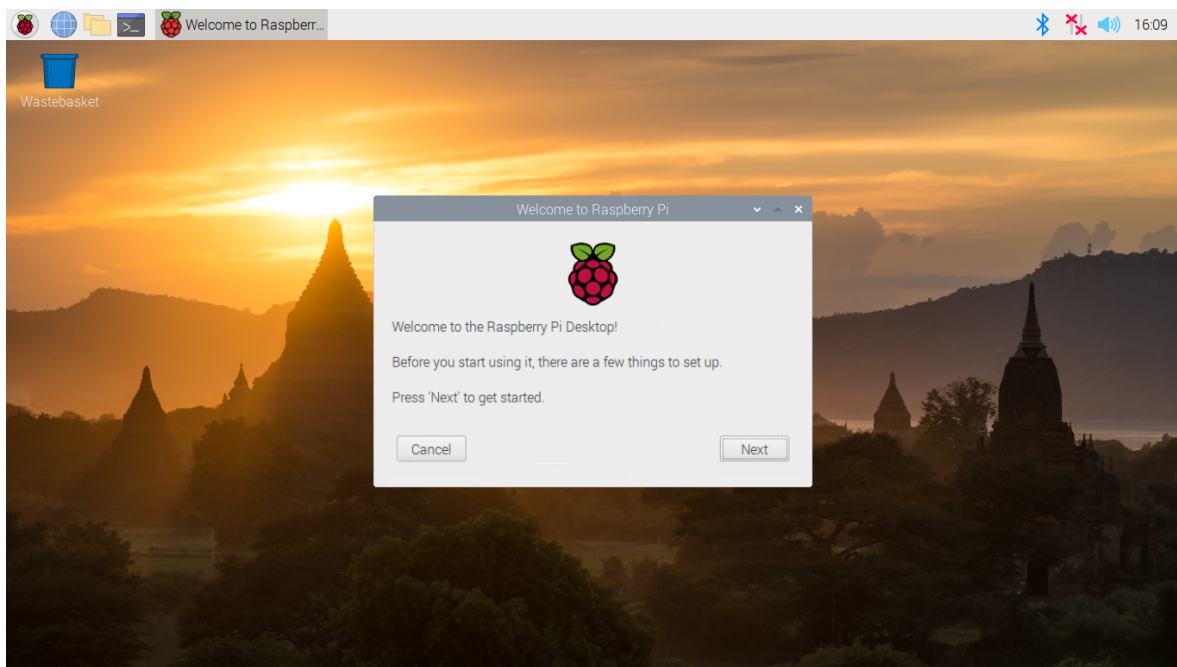


Figure 1: Welcome screen after booting your Raspberry pi device for the first time

---

<sup>2</sup><https://www.raspberrypi.org/software/>

## 2.2 Raspberry Pi quick start

Now we have all the hardware and software in place and ready to start. Booting your Raspberry pi device for the first time, you can go through initial setup of the device. Connect your device to a Wi-Fi network and be sure to change the default password for your device. It is also recommended to update your software.

After completing the setup open the terminal and type:

```
$ sudo raspi-setup
```

This command should open the software configuration tool like in figure 2.

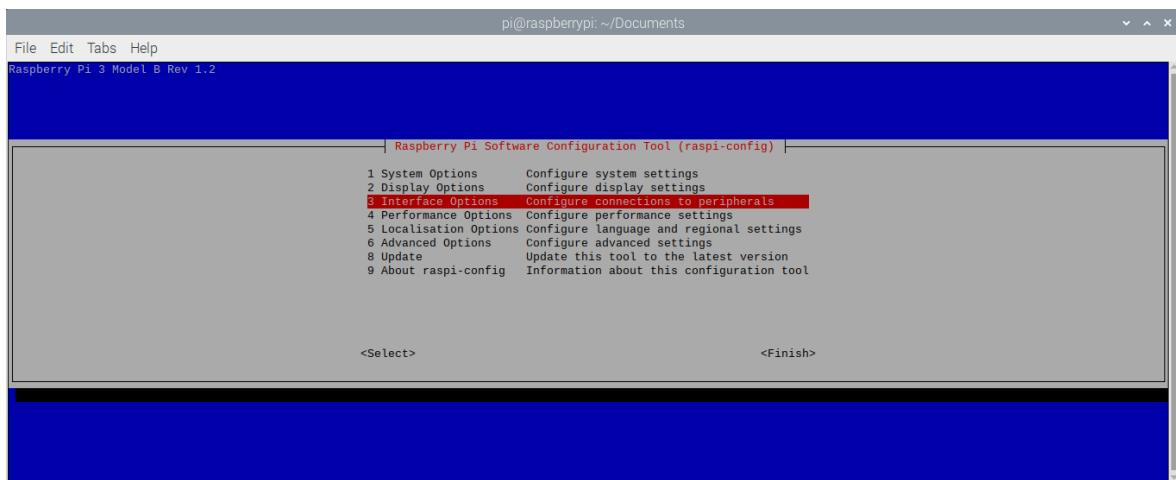


Figure 2: Raspberry pi software configuration tool

From here you can enable the camera interface by navigating through **Interface Options**>**Camera** and choose **Enable**. Do the same for **SSH** and **I2C**.

From now on you can access the Raspberry Pi remotely. I am using Putty<sup>3</sup> as the SSH utility to connect to the Raspberry Pi shell. Returning back to the terminal you can issue the

```
$ ifconfig
```

command to see the IP address of your Raspberry Pi. You will connect to this IP address in Putty to connect to the Raspberry Pi (if you have not changed the default password you can use pi as the login name and raspberry as the password.)

---

<sup>3</sup><https://www.putty.org/>

## 3 Create Azure resources

IoT Edge runtime, we will be installing on Raspberry Pi in the next section, will communicate to an IoT Hub in Azure. This IoT Hub needs to be created and initialized in Azure platform as well as other resources needed for this solution. It is possible to create these resources on the Azure portal or through command line tool package, Azure CLI, provided by Microsoft. Here we will use the Azure portal.

### 3.1 Create an IoT hub

This section describes how to create an IoT Hub using the Azure portal:

1. Sign in to the Azure portal
2. From the Azure homepage, select the + **Create a resource** button, and then enter *IoT Hub* in the **Search the Marketplace** field,
3. Select **IoT Hub** from the search results, and then select **Create**.
4. On the **Basics** tab, complete the fields as follows:
  - **Resource Group:** Select a resource group or create a new one you will use for this project. To create a new one, select **Create new** and fill in the name you want to use. To use an existing resource group, select that resource group.
  - **Region:** Select the region in which you want your hub to be located. Select the location closest to you.
  - **IoT Hub Name:** Enter a name for your hub. This name must be globally unique, with a length between 3 and 50 alphanumeric characters. The name can also include the dash ('-') character.  
(Because the IoT hub will be publicly discoverable as a DNS endpoint, be sure to avoid entering any sensitive or personally identifiable information when you name it.)
5. Select **Next: Networking** to continue creating your hub.

Choose the endpoints that devices can use to connect to your IoT Hub. You can select the default setting **Public endpoint (all networks)**, or choose **Public endpoint (selected IP ranges)**, or **Private endpoint**.

Accept the default setting for this project.

6. Select **Next: Management** to continue creating your hub.

- **Pricing and scale tier:** Your selected tier. You can choose from several tiers, depending on how many features you want and how many messages you send through your solution per day. The IoT Edge feature we need for this project is included in the Standard tier. For this project you can use the Free Standard Tier.
- **IoT Hub units:** For this project we need only one unit.
- **Defender for IoT:** Turn this on to add an extra layer of threat protection to IoT and your devices. This option is not available for hubs in the free tier. If you've chosen one of the paid tiers, can leave this option off.

7. Select **Next: Tags** to continue to the next screen.

Tags are name/value pairs. You can assign the same tag to multiple resources and resource groups to categorize resources and consolidate billing. In this project, you won't be adding any tags.

8. Select **Next: Review + create** to review your choices. You see something similar to this screen, but with the values you selected when creating the hub.

9. Select **Create** to start the deployment of your new hub. Your deployment will be in progress a few minutes while the hub is being created. Once the deployment is complete, click **Go to resource** to open the new hub.

### 3.1.1 Register an IoT Edge device in IoT Hub

This subsection provides the steps to register a new IoT Edge device in your IoT Hub.

In your IoT hub in the Azure portal, IoT Edge devices are created and managed separately from IoT devices that are not edge enabled.

- Start by navigating to your IoT Hub.
- In the left pane, select **IoT Edge** from the menu, then select **Add an IoT Edge device**.
- On the **Create a device** page, provide the following information:
  - Create a descriptive device ID.
  - Select **Symmetric key** as the authentication type.

- Use the default settings to auto-generate authentication keys and connect the new device to your hub.
- Select Save.

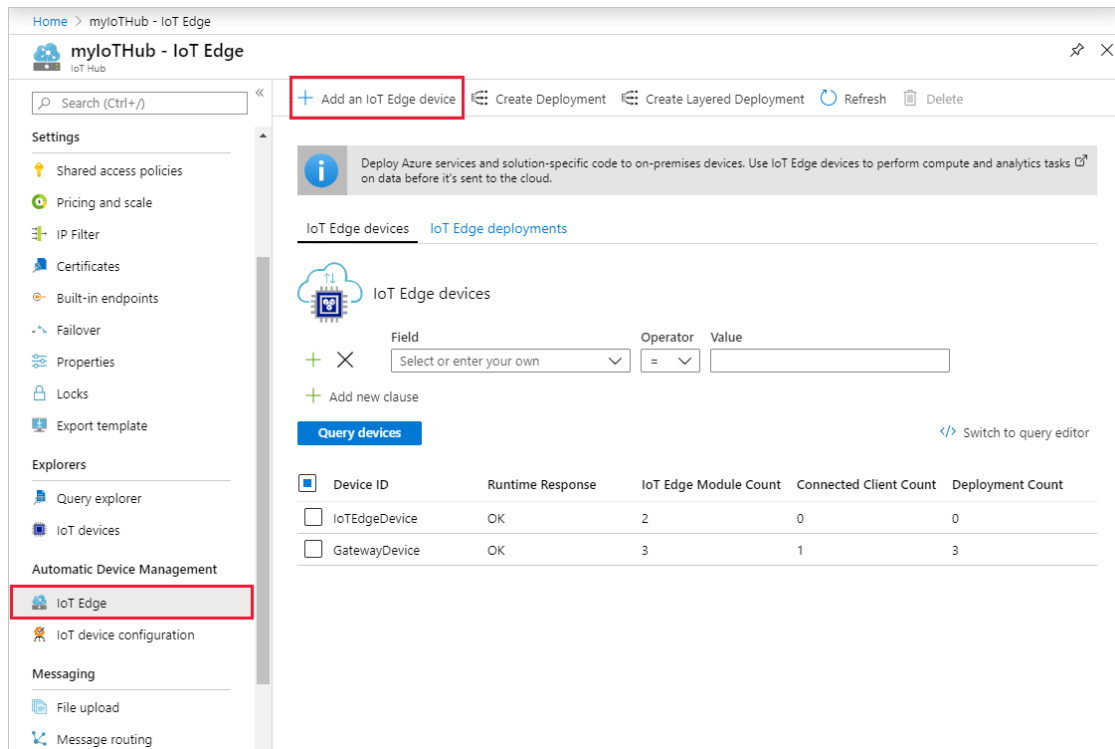


Figure 3: Add IoT Edge Device

Now that you have a device registered in IoT Hub, retrieve the *Primary Connection String* which you will use to complete installation and provisioning of the IoT Edge runtime.

Devices that authenticate with symmetric keys have their connection strings available to copy in the portal:

- From the **IoT Edge** page in the portal, click on the device ID from the list of IoT Edge devices.
- Copy the value of **Primary Connection String**.



## 3.2 Create a Storage Account for Table Storage

An Azure storage account contains all of your Azure Storage data objects: blobs, file shares, queues, tables, and disks. The storage account provides a unique namespace for your Azure Storage data that's accessible from anywhere in the world over HTTP or HTTPS. Data in your storage account is durable and highly available, secure, and massively scalable.

We will use our Storage Account to store data sent from the Edge device.

The following steps describe how to create a Storage Account for Table Storage using the Azure portal:

- From the upper left-hand corner of the Azure portal, select **Create a resource > Storage > Storage account**.
- Choose **Classic** for the deployment model and click on **Create**.
- Enter the name of your choice for the account name, Standard for the type, choose your subscription, select the resource group you created earlier. Choose the same location and resource group as the IoT Hub you created. Then click on **Review + Create** to create the account.

Once the account is created, find it in the **resources blade**, go to **Settings, Keys**, and write down the *primary connection string*.

## 3.3 Create a Stream Analytics job to Save Data in Table Storage

Stream Analytics is an Azure IoT service that streams and analyzes data in the cloud. We'll use it to process data coming from your device.

- Select **Create a resource** in the upper left-hand corner of the Azure portal.
- Select **Analytics > Stream Analytics job** from the results list.
- Fill out the Stream Analytics job page. Enter a name for the job, a preferred region, then choose your subscription. At this stage you are also offered to create a new or to use an existing resource group. Choose the resource group you created earlier.

### 3.3.1 Configure job input

In this section, you will configure an IoT Hub device input to the Stream Analytics job. Use the IoT Hub you created in the previous section.

- Navigate to your Stream Analytics job.
- Select **Inputs > Add Stream input > IoT Hub**.
- Fill out the **IoT Hub** page with the following values:
  - **Input Alias**: name your input alias (in my example i'm using "OdlukaInput")
  - **Source Type**: Data Stream
  - **Source**: IoT Hub
  - **Subscription**: Choose your subscription
  - **IoT Hub**: use the name for the IoT Hub you create before
  - **Shared Access Policy Name**: Shared access policies: iothubowner
  - **Shared Access Policy Key**: The iothubowner primary key can be found in your IoT Hub **Settings > Shared access policies**
  - IoT Hub Consumer Group: leave it to the default value
  - Event serialization format: JSON
  - Encoding: UTF-8

### 3.3.2 Configure job output

Navigate to the Stream Analytics job that you created earlier. Click on the **Outputs** tile and in the **Outputs blade**, click on **Add > Table storage**. Enter the following settings then click on **Create**:

- **Output Alias**: OdlukaTable
- **Sink**: Table Storage
- **Storage account**: The storage you made earlier
- **Storage account key**: The primary key from the storage account you made earlier (*can be found in Settings > Keys > Primary Access Key*)

- **Table Name:** TableOdluka1 (You can choose what ever name you want - If the table doesn't already exist, Local Storage will create it)
- **Partition Key:** DeviceId
- **Row Key:** created
- **Batch size:** 1

### 3.3.3 Define the transformation query

Navigate to the Stream Analytics job that you created earlier. Click on the **Query tile** (next to the Inputs tile). In the Query settings blade, type in the below query and click **Save**:

Listing 1: Stream Analytics Job Query

```

SELECT
    prediction.ArrayValue.tagName AS Name,
    prediction.ArrayValue.probability AS Probability ,
    OdlukaInput.created ,
    OdlukaInput.IoTHub.ConnectionDeviceId AS DeviceID
INTO
    OdlukaTable
FROM
    OdlukaInput TIMESTAMP BY OdlukaInput.created
CROSS APPLY
    GetArrayElements(OdlukaInput.prediction) AS prediction
WHERE
    prediction.ArrayValue.probability > 0.8

```

Here, my input alias is "OdlukaInput" and my output alias is "OdlukaTable." If you've chosen other input and output aliases, use those names in the query instead.

Back in the Stream Analytics blade, start the job by clicking on the **Start** button at the top.

**Note:** You are charged for this resource for as long as the job is running so be sure to *Stop* every Stream Analytics Job when you're not using it!

### 3.3.4 Create a Container Registry

Azure Container Registry is a private registry service for building, storing, and managing container images and related artifacts. In this project, you create an Azure container registry instance with the Azure portal. Later we will push container images we need for our project into the registry, and finally pull and run the image from your registry onto the device.

Follow the next steps to create a container registry:

- Select **Create a resource** in the upper left-hand corner of the Azure portal.
- Select **Containers > Container Registry**.
- On the **Basics** tab, complete the fields as follows:
  - **Subscription**: Select your subscription. If you have multiple, select the one you've been using for other resources.
  - **Resource Group**: Choose the resource group you created earlier.
  - **Location**: Select the location closest to you or the one you used for other resources like your IoT Hub.
  - **Registry name**: You can choose whatever name you want. The registry name must be unique within Azure, and contain 5-50 alphanumeric characters.
  - **SKU**: Select 'Basic'.

Accept default values for the remaining settings. Then select **Review + create**. After reviewing the settings, select **Create**.

When the **Deployment succeeded** message appears, select the container registry in the portal.

Take note of the registry name and the value of the **Login server**, which is a fully qualified name ending with *azurecr.io* in the Azure cloud. You will use these values in the later steps when you push and pull the images from the container.

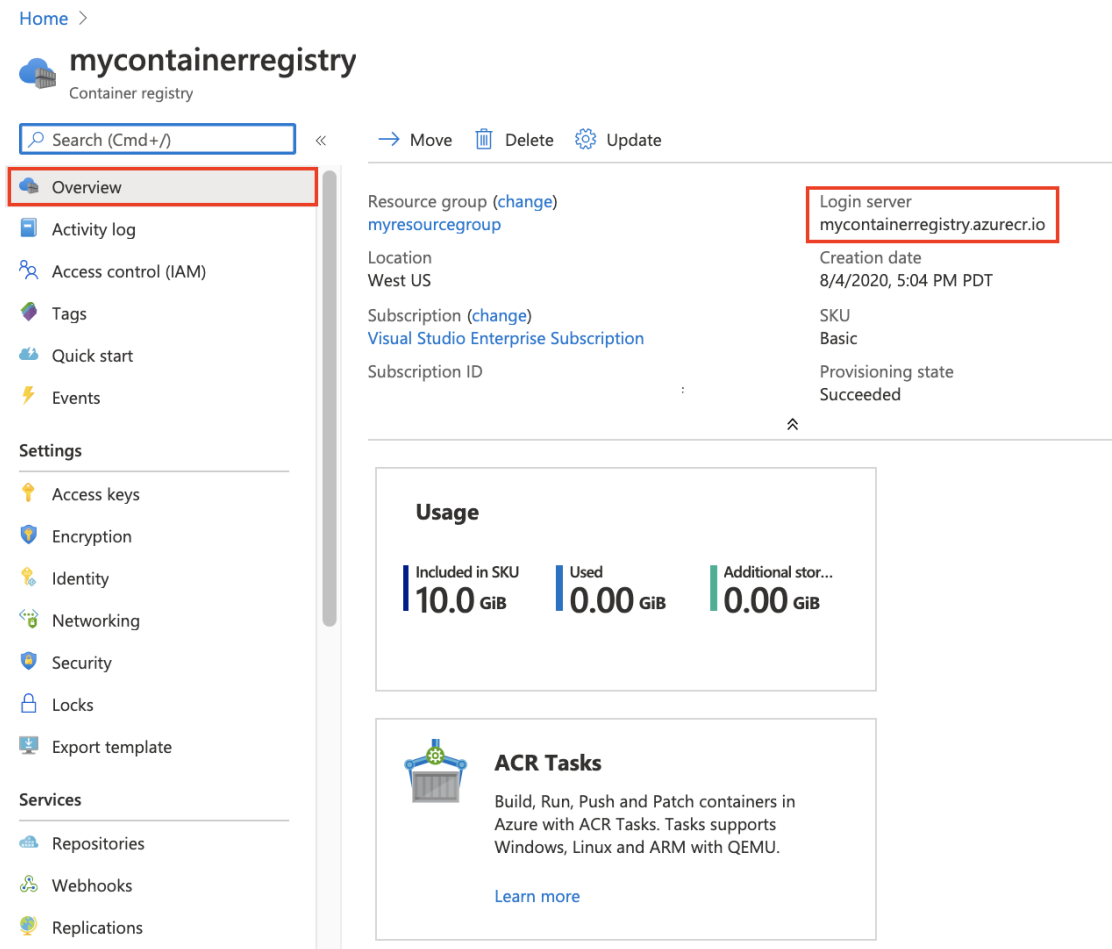


Figure 4: Container Registry Overview

## 4 IoT Edge on Raspberry Pi

This section will cover the installation of Azure IoT Edge runtime on your Raspberry Pi device as well as connecting the Edge device to the IoT Hub in Azure.

### 4.1 SSH into your Raspberry Pi

Now that you have enabled SSH and found out your IP address you can go ahead and SSH into your Raspberry Pi from any other computer. You'll also need the username and the password for the Raspberry Pi.

Default Username and Password are:

- username: pi
- password: raspberry

If you have changed the default password then use the new password instead of the above.

On a successful login you'll be presented with the terminal of your Raspberry Pi. Now you can run any commands on your Raspberry Pi through this terminal remotely (within the current network) without having to access your Raspberry Pi physically.

## 4.2 Install IoT Edge runtime on Raspberry Pi

The Azure IoT Edge runtime is what turns a device into an IoT Edge device. Once a device is configured with the IoT Edge runtime, you can start deploying business logic to it from the cloud. This section lists the steps to install the Azure IoT Edge runtime on your Raspberry Pi device.

Enter the following command into the Raspberry pi terminal you have opened:

```
curl https://packages.microsoft.com/config/debian/stretch\
/multiarch/prod.list > ./microsoft-prod.list

sudo cp ./microsoft-prod.list /etc/apt/sources.list.d/

curl https://packages.microsoft.com/keys/microsoft.asc | \
gpg --dearmor > microsoft.gpg

sudo cp ./microsoft.gpg /etc/apt/trusted.gpg.d/
```

Now you have prepared your device to access the Microsoft installation packages.

With the following commands you will have a container engine needed for Azure IoT Edge runtime:

```
sudo apt-get update
sudo apt-get install moby-engine
```

Now we have all we need to install IoT Edge on your device:

```
sudo apt-get install iotedge=1.1* libiothsm-std=1.1*
```

The IoT Edge security daemon provides and maintains security standards on the IoT Edge device. The daemon starts on every boot and bootstraps the device by starting the rest of the IoT Edge runtime.

### 4.3 Provision the device with its cloud identity

Now that the container engine and the IoT Edge runtime are installed on your device, you're ready for the next step, which is to set up the device with its cloud identity and authentication information.

On the IoT Edge device, open the configuration file:

```
sudo nano /etc/iotedge/config.yaml
```

Find the provisioning configurations of the file and uncomment the **Manual provisioning configuration using a connection string** section, if it isn't already uncommented.

```
# Manual provisioning configuration using a connection string
provisioning:
  source: "manual"
  device_connection_string: "<ADD DEVICE CONNECTION STRING HERE>"
```

Update the value of **device\_connection\_string** with the primary connection string from your IoT Edge device in the cloud. Make sure that any other provisioning sections are commented out. Make sure the **provisioning:** line has no preceding whitespace and that nested items are indented by two spaces. (*To paste clipboard contents into Nano Shift+Right Click or press Shift+Insert.*)

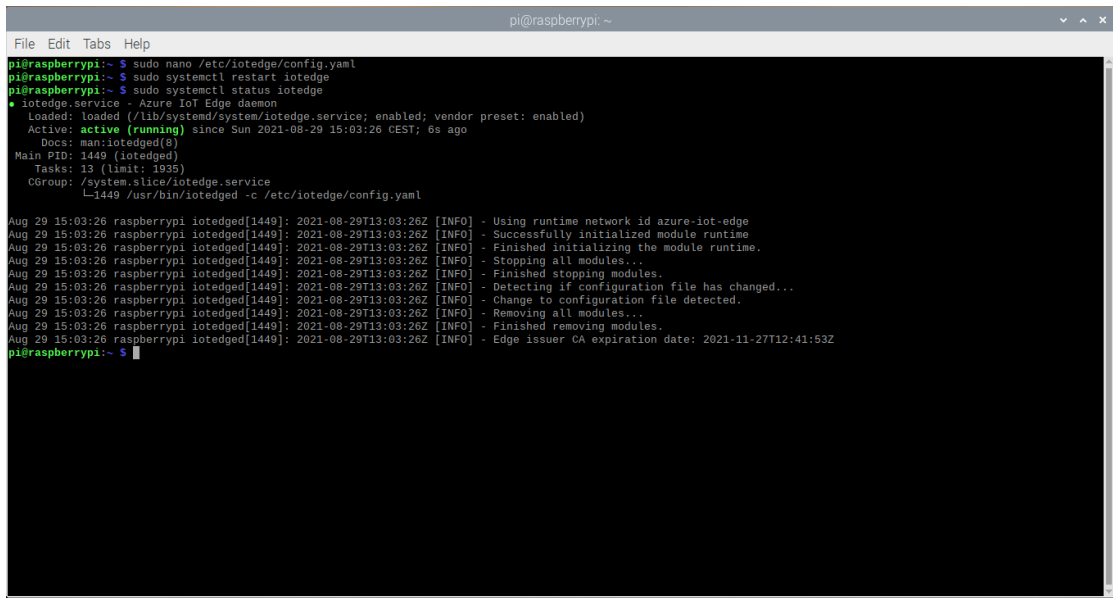
Save and close the file. (*CTRL + X, Y, Enter*)

After entering the provisioning information in the configuration file, restart the daemon:

```
sudo systemctl restart iotedge
```

Verify that the runtime was successfully installed and configured on your IoT Edge device with:

```
sudo systemctl status iotedged
```



```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~$ sudo nano /etc/iotedge/config.yaml  
pi@raspberrypi:~$ sudo systemctl restart iotedged  
pi@raspberrypi:~$ sudo systemctl status iotedged  
● iotedged.service - Azure IoT Edge daemon  
   Loaded: loaded (/lib/systemd/system/iotedged.service; enabled; vendor preset: enabled)  
   Active: active (running) since Sun 2021-08-29 15:03:26 CEST; 6s ago  
     Docs: man:iotedged(8)  
   Main PID: 1449 (iotedged)  
    Tasks: 13 (limit: 1939)  
   CGroup: /system.slice/iotedged.service  
           └─1449 /usr/bin/iotedged -c /etc/iotedge/config.yaml  
  
Aug 29 15:03:26 raspberrypi iotedged[1449]: 2021-08-29T13:03:26Z [INFO] - Using runtime network id azure-iot-edge  
Aug 29 15:03:26 raspberrypi iotedged[1449]: 2021-08-29T13:03:26Z [INFO] - Successfully initialized module runtime  
Aug 29 15:03:26 raspberrypi iotedged[1449]: 2021-08-29T13:03:26Z [INFO] - Finished initializing the module runtime.  
Aug 29 15:03:26 raspberrypi iotedged[1449]: 2021-08-29T13:03:26Z [INFO] - Stopping all modules...  
Aug 29 15:03:26 raspberrypi iotedged[1449]: 2021-08-29T13:03:26Z [INFO] - Finished stopping modules.  
Aug 29 15:03:26 raspberrypi iotedged[1449]: 2021-08-29T13:03:26Z [INFO] - Detecting if configuration file has changed...  
Aug 29 15:03:26 raspberrypi iotedged[1449]: 2021-08-29T13:03:26Z [INFO] - Change to configuration file detected.  
Aug 29 15:03:26 raspberrypi iotedged[1449]: 2021-08-29T13:03:26Z [INFO] - Removing all modules...  
Aug 29 15:03:26 raspberrypi iotedged[1449]: 2021-08-29T13:03:26Z [INFO] - Finished removing modules.  
Aug 29 15:03:26 raspberrypi iotedged[1449]: 2021-08-29T13:03:26Z [INFO] - Edge issuer CA expiration date: 2021-11-27T12:41:53Z  
pi@raspberrypi:~$
```

Figure 5: systemctl status iotedged

Use the check tool to verify configuration and connection status of the device.

```
sudo iotedged check
```

View all the modules running on your IoT Edge device. When the service starts for the first time, you should only see the **edgeAgent** module running. The edgeAgent module runs by default and helps to install and start any additional modules that you deploy to your device.

```
sudo iotedged list
```

When you create a new IoT Edge device, it will display the status code 417 – The device's deployment configuration is not set in the Azure portal. This status is normal, and means that the device is ready to receive a module deployment.



## References

- [1] Microsoft. Azure container registry documentation. <https://docs.microsoft.com/en-us/azure/container-registry/>.
- [2] Microsoft. *Azure IoT Edge documentation*.
- [3] Microsoft. Azure iot hub pricing. <https://azure.microsoft.com/en-us/pricing/details/iot-hub/>.
- [4] Microsoft. Install or uninstall azure iot edge for linux. <https://docs.microsoft.com/en-us/azure/iot-edge/how-to-install-iot-edge?view=iotedge-2018-06>.
- [5] Microsoft. Register an iot edge device in iot hub. <https://docs.microsoft.com/en-us/azure/iot-edge/how-to-register-device?view=iotedge-2018-06>.
- [6] Microsoft. Understand iot edge automatic deployments. <https://docs.microsoft.com/en-us/azure/iot-edge/module-deployment-monitoring?view=iotedge-2020-11>.
- [7] Raspberry Pi. Setting up your raspberry pi. <https://www.raspberrypi.org/documentation/computers/getting-started.html>.