Azure Edge Installer User Guide

Version: 1.2.2 8/19/2022

Contents

Overview

The Azure Edge Installer is an onboarding tool that streamlines the provisioning process for edge devices. Developers can use the Installer to register their device to Azure IoT Hub and install core components for securing and managing intelligent edge devices.

The Installer offers the following benefits:

1. Accelerates device onboarding to Azure
2. Installs base components for manageability and security
3. Establishes a device setup that can be scaled to production scenarios
4. Configures devices to begin Edge AI development

Hardware Options

The Installer officially supports the following devices:

- NVIDIA AGX Orin/Orin NX
- NVIDIA AGX Xavier/Xavier NX (only on Jetpack 5.0)

However, the Installer should work on other ARM64 devices with Ubuntu 20.0.4. If your device meets these requirements, feel free to try the Installer and report any feedback to our team.

Device Setup (for NVIDIA Xavier/Orin Devices)

For NVIDIA Jetson devices, you will need the following equipment:

- External monitor + USB keyboard/mouse
- **[Microsoft corporate network]** Ethernet cable recommended
- **[Orin Only]** DP-to-DP cable and monitor with display port (DP-to-HDMI cable will not work)
- **[JP4.x OS/FW Only]** Additional physical host machine with Ubuntu OS (This is only required if your developer kit currently boots with JP4.x OS/FW(Ubuntu 18.04))
    - Please see Device Setup for more information

If you are setting up your device for the first time, refer to this guidance:

- **Orin**: See Getting Started with Jetson Orin guide.
- **Xavier**: See Getting Started With Jetson Xavier NX guide. Note that you will need to flash Jetpack 5.0 or later for your device to be compatible with Edge Installer.

[Important] When going through the Jetson device setup, make sure you include not only the Jetson Linux BSP but also the **JetPack SDK** if you plan to deploy DeepStream workloads to the device.

If your DK is already setup but not registered to Azure, refer to this guidance:

1. Check the OS of your device by running command "***lsb_release -a***". Orin devices will likely be Ubuntu 20.0.4, which is sufficient for the Installer. Xavier devices will likely be Ubuntu 18.0.4, which needs to be upgraded to use the Installer.
2. For devices on ubuntu 18.04: You will need to flash Jetpack 5.0 or later for your device to be compatible with Edge Installer. Navigate to NVIDIA's guidance for flashing your specific device.
3. For devices on ubuntu 20.04: You can proceed to run the installer. But it is recommended to install the latest version of Jetpack 5.0 by following Step 2 of Getting Started with Jetson Orin guide

If your DK is already registered to an IoT Hub/Edge account, you can still follow the guidance of using connection string to run the installer and connect to the same IoTHub. The Installer should effectively install Defender for IoT and OSConfig.

How to Install

1. **Make sure the developer kit is connected to the internet before executing the installer!**

2. Set up Azure Basics
    a. If you do not have an IoT Hub, follow "Create an IoT Hub" in Use the Azure portal to create an IoT Hub | Microsoft Docs. You can skip this step if you already have an IoT Hub.
    b. Once you have an IoT Hub, choose one of these two options:
        - **[Option I] Provision with connecting string**
          If you are not familiar with the process of getting the IoT Hub connection string. Refer to the steps in appendix: *Provision with connecting string.*
        - **[Option II] Provision with DPS**
          Refer to the steps in appendix: *How to setup IoT Hub with DPS.*
3. Save the **connection string** to a .txt file (or if using DPS provisioning, save Registration_ID, Symmetric Primary Key, the IoT Hub host name, DPS Scope ID to a .txt file).
4. Download the installer .run file (ex: **eai-installer_1.2.2_arm64.run**)
5. Copy the installer file and the txt file that contains provision info to your NV developer kit by USB drive
    - **[Alternate Method]** If you do not have a USB drive but the DK and PC are connected to the same network, there is a work around. Navigate to the PC command line, enter the directory where the installer is located, and run the following command to securely copy the installer to your DK.

      ```
      scp ~/Downloads/eai-installer_[VERSION]_arm64.run [DK desktop address]:~/[target directory]./e
      ```

6. Before running the installer, execute the following commands in the **command line of your DK.**

   From the directory of the installer .run file, execute the following command:

   ```
   sudo chmod +x eai-installer_1.2.2_arm64.run; sudo ./eai-installer_1.2.2_arm64.run; cd /usr/local/microsoft/eai-installer
   ```

   [Note] 1.2.2 is the current installer version, if you are using other versions, please modify the command accordingly.

   [Optional] Use the following command if you want to verify the installer is correctly setup.

   ```
   dpkg-query -s eai-installer
   ```

7. To execute the installer and set the developer kit Azure ready, you can choose one of the provisioning mechanisms below when executing the installer. Make sure you are under the directory of *"/usr/local/microsoft/eai-installer"* when executing the following command.

   [Connection String]

   ```
   sudo ./azure-iot-edge-installer.sh -c  "<Azure IoT Edge Device Connection String>"
   ```

   [DPS]

   ```
   sudo ./azure-iot-edge-installer.sh -s <ID Scope> -r <Registration ID> -k <Symmetric Key>
   ```

[Note] To disable the telemetry sending to Microsoft, add the parameter **-nt** or **–telemetry-opt-out** when executing the shell script.

8. Refer to the help menu for more options:

```
sudo ./azure-iot-edge-installer.sh -h
```

Post-Installer Experience

Using the Services in the Fundamentals Package

- **Defender for IoT**
    1. To configure the Defender for IoT agent-based solution, please follow "Configure data collection" section and the "Log Analytics creation".
    2. Navigate to **IoT Hub** > **Your hub** > **Defender for IoT** > **Overview** and review the information. Refer to the following guidance for detail.
        - Investigate security recommendations
        - Investigate security alerts
    3. Refer to Micro agent event collection (Preview) for more applications with the Defender for IoT.
- **OSConfig**

  Refer to the following guidance to try the OSconfig features. (More scenarios are available under the same category: "*What can I provision and manage*".)
    - Working with host names using Azure IoT and OSConfig
    - Reboot or shut down devices with Azure IoT and OSConfig

Updating Packages

The installer installs specific versions of Microsoft fundamental packages (IoTEdge, OSConfig, Defender for IoT). This is the verified known stable combination. Please refer to the release note for versions information. If for any reason you want to use the latest fundamental packages, run the following command AFTER you have successfully run the installer. This will trigger the update of those components.

```
sudo ./azure-iot-edge-installer.sh -u
```

Uninstallation

1. Uninstall Edge runtime & Percept packages

   $ sudo apt-get remove --purge aziot-edge aziot-identity-service osconfig eai-installer -y
2. Enumerate and remove edge modules

   $ sudo docker ps -a

   $ sudo docker rm -f <container id>

   $ sudo docker images

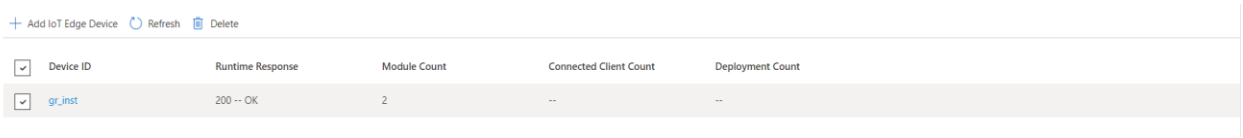
   $ sudo docker rmi -f <image id>

4. Restart your device
   $ sudo reboot

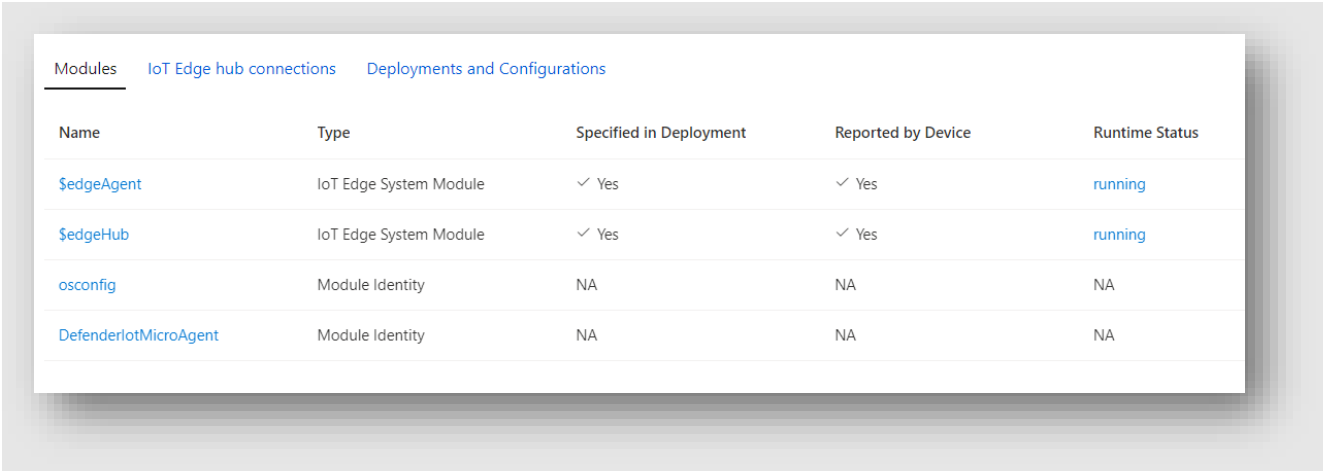
Troubleshooting

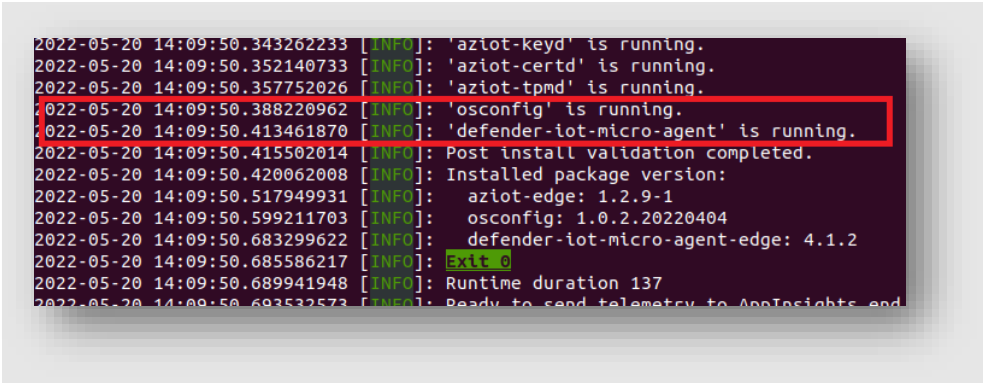
Edge runtime and module status

The expected status of IoT Edge runtime response is **“NA”** or **“417 – The device’s deployment configuration is not set”**. It will change to **“200 – OK”** after the first module deployment. (You can simply click the “Set Modules” to trigger an empty deployment.)



For **DefenderMicroAgent** and **OSConfig** module, it is expected to see the states show **“NA”**. As long as you get the following INFO notification from the installer output, you are good to go.





Configuration Check

The IoT Edge Check is a useful tool for checking the status of edge agent and edge hub. Before you proceed to the check process, please double confirm if system time has been synchronized after network configured.

```
Sudo timedatectl status
```

Then, you can perform configuration and connectivity checks.

```
Sudo iotedge check
```

Internal DNS (Highly Recommended)

If your network environment is under corpnet, you may need to follow the steps below to add an internal DNS server for Docker network to avoid IoTEdge connectivity issue.
(follow the instruction carefully, or refer to VI Editor with Commands in Linux/Unix Tutorial to learn details about editing file in terminal.)

1. When the device is already connected to the corpnet, use the following command to check the DNS server IP.
   $ sudo systemd-resolve --status
2. In a terminal window, us the following command to open the config file:
   $ sudo vi /etc/docker/daemon.json
3. Find the following line and move the cursor to the highlighted location. Then press "**i**" to enter insert mode.
   "dns": ["1.1.1.1", "8.8.8.8"],
4. Insert the characters so the line is modified as below. (Your DNS server IP needs to be the fist element in the array.)
   "dns": [**"<DNS server IP>",** "1.1.1.1", "8.8.8.8"],
5. Press **Esc** key to exit the insert mode.
6. Type "**:wq**" to save and exit.
7. Restart Docker service after the configuration is updated.
   $ sudo systemctl restart docker

Further IoT Edge diagnostic

For information about each of the diagnostic checks this tool runs, including what to do if you get an error or warning, see IoT Edge troubleshoot checks. The Edge Installer performs the installation and default setup process. However, there are other factors that impact the behavior of IoT Edge runtime/service (such as network configuration, firewall…). Therefore, please do check the following IoT Edge troubleshooting guidance if you encounter IoT Edge issues:

- Common errors – Azure IoT Edge | Microsoft Docs
- Troubleshoot – Azure IoT Edge | Microsoft Docs
- Troubleshoot from the Azure portal – Azure IoT Edge | Microsoft Docs

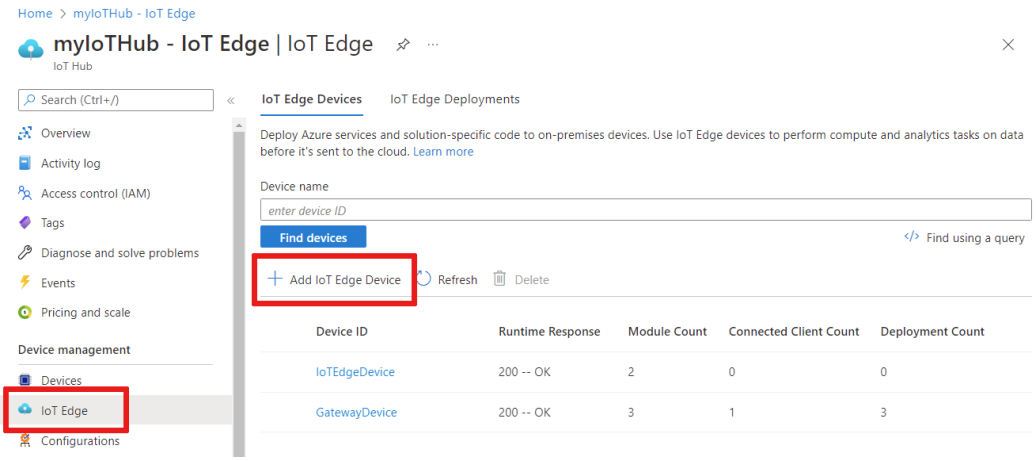
[Note] DPS provisioning does not contain IoTHub's hostname, so users need to add additional parameter for iotedge check command. Refer to the following links for troubleshooting:

- [iotedge check with DPS & x.509 configuration returns "cannot resolve IoT Hub hostname" Errors · Issue #2033 · Azure/iotedge (github.com)]
- [iotedge check incorrectly shows an error when using DPS · Issue #2313 · Azure/iotedge (github.com)]

Appendix

Provision with connecting string

1. **Register your device**
   In your IoT hub in the Azure portal, IoT Edge devices are created and managed separately from IoT devices that are not edge enabled.
   a. Sign in to the Azure portal and navigate to your IoT hub.
   b. In the left pane, select **IoT Edge** from the menu, then select **Add an IoT Edge device**.

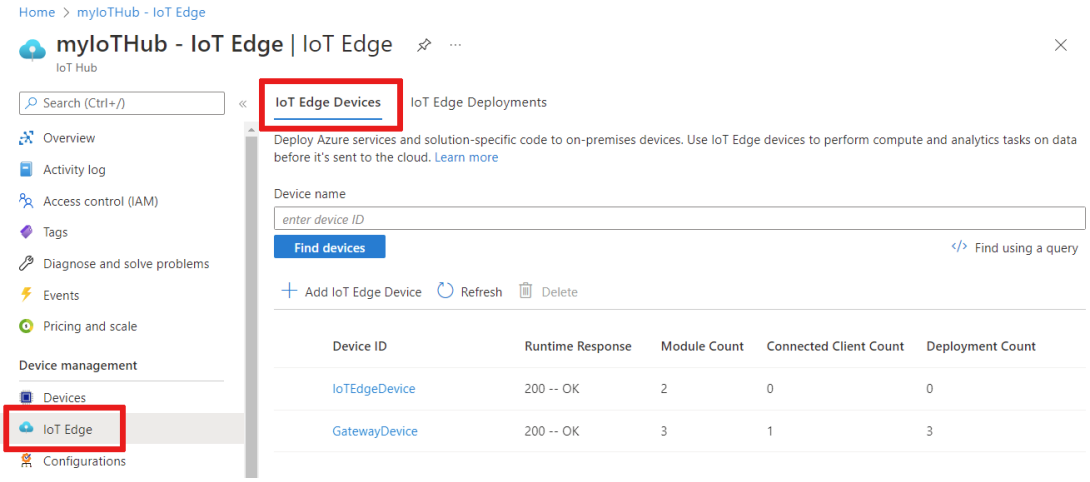


   c. On the Create a device page, provide the following information:
      1. Create a descriptive device ID. Make a note of this device ID, as you'll use it later.
      2. Select **Symmetric key** as the authentication type.
      3. Use the default settings to auto-generate authentication keys and connect the new device to your hub.
   d. Select **Save**.

   Now that you have a device registered in IoT Hub, retrieve the information that you use to complete installation and provisioning of the IoT Edge runtime.

2. **View registered devices and retrieve provisioning information**
   Devices that use symmetric key authentication need their connection strings to complete installation and provisioning of the IoT Edge runtime.

All the edge-enabled devices that connect to your IoT hub are listed on the IoT Edge page.



When you're ready to set up your device, you need the connection string that links your physical device with its identity in the IoT hub.
Devices that authenticate with symmetric keys have their connection strings available to copy in the portal.

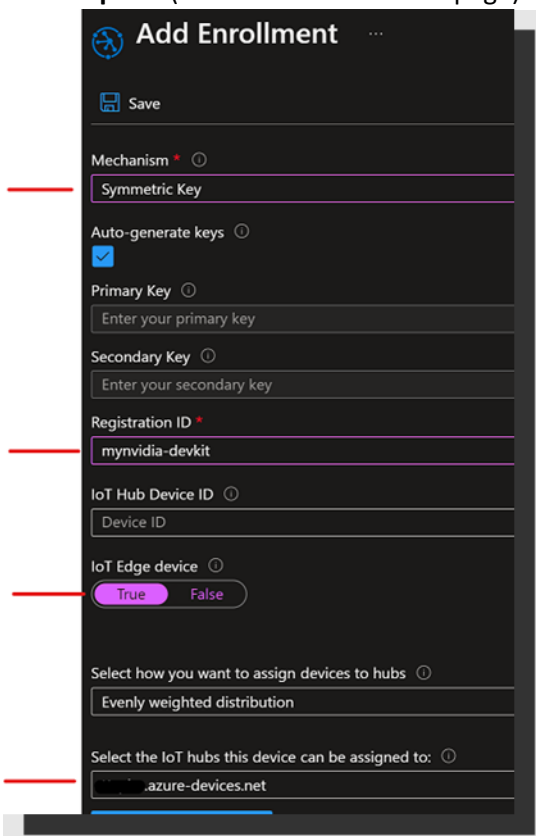
1. From the **IoT Edge** page in the portal, click on the device ID from the list of IoT Edge devices.
2. Copy the value of either **Primary Connection String** or **Secondary Connection String**.

Refer to [Create and provision an IoT Edge device on Linux using symmetric keys - Azure IoT Edge | Microsoft Docs] for more details.

How to setup IoT Hub with DPS

1. Set-Up Azure Basics – If you do not have an IoT Hub (and DPS linked to your IoT Hub), follow these instructions to create an IoT Hub, create a DPS instance, and link the IoT Hub to your DPS instance
2. Create an individual enrollment within the DPS resource
    a. Specify the Mechanism to be **Symmetric Key**, and set **IoT Edge Device = TRUE**. Refer to the image below for filling in remaining fields.
    b. Copy the following for later use.
        i. the **Registration_ID**,
        ii. **Symmetric Primary Key**,
        iii. the **IoT Hub host name** that's been assigned.

iv. **DPS Scope ID** (found in DPS overview page)



c. [Note] If the dropdown menu from the final field in the image above does not include an IoT Hub, still proceed to the next page – where the option will likely update.
d. [Note] The Installer does not support the following characters for DeviceID: =, %, !, $
e. [Note] If you would like to utilize the hostname option (**-hn** or **--hostname**) to assign hostname as *DeviceId*, please also notice the following special characters do not comply with RFC 1035 for hostname naming: + _ # * ? ( ) , : @ '