**Complete Offline Installation of Azure IoT Edge**

The purpose of this document is to outline the steps necessary to create an offline installation of Azure IoT Edge. Many of these steps are dependent on the Linux distribution and version you plan to deploy to. However, the general steps can be applied to any distribution and version. The examples use Ubuntu version 18.04. You will need to substitute the appropriate files for your distribution and version in the instructions. The instructions demonstrate the setup of IoT Edge devices using symmetric key authentication. Please see the Azure IoT Edge documentation for further details regarding device registration using X.509 certificate or Trusted Platform Module (TPM) authentication if you are using one of these device authentication mechanisms.

**Prerequisites**

* To gather the files needed for offline Azure IoT Edge installation you will need a device capable of downloading files from the internet with Docker installed.
* Follow the instructions in the section labeled Create a hierarchy of IoT Edge devices in this document [Tutorial - Create a hierarchy of IoT Edge devices - Azure IoT Edge | Microsoft Docs](https://docs.microsoft.com/en-us/azure/iot-edge/tutorial-nested-iot-edge?view=iotedge-2020-11#create-a-hierarchy-of-iot-edge-devices). This will ensure you have the configuration files necessary to configure your IoT Edge device(s).

**Gather the deployment files**

The following set of steps need to be executed on a device with internet access.

1. Download the Microsoft package repository

wget https://packages.microsoft.com/config/ubuntu/18.04/multiarch/packages-microsoft-prod.deb -O packages-microsoft-prod.deb

1. Download a container engine (choose either Docker or Moby)
   1. Docker
      1. While Moby is the officially supported container engine for Azure IoT Edge, Docker can be used as well. Docker provides excellent instructions for downloading and installing their container engine in an offline scenario as outlined below.

wget https://download.docker.com/linux/ubuntu/dists/bionic/pool/stable/amd64/containerd.io\_1.4.9-1\_amd64.deb -O containerd.io.deb

wget https://download.docker.com/linux/ubuntu/dists/bionic/pool/stable/amd64/docker-ce-cli\_20.10.12~3-0~ubuntu-bionic\_amd64.deb -O docker-ce-cli.deb

wget https://download.docker.com/linux/ubuntu/dists/bionic/pool/stable/amd64/docker-ce\_20.10.12~3-0~ubuntu-bionic\_amd64.deb -O package.deb

* 1. Moby
     1. Moby packages for different distributions and versions can be found at packages.microsoft.com.

wget https://packages.microsoft.com/ubuntu/18.04/prod/pool/main/m/moby-runc/moby-runc\_1.0.3+azure-1\_amd64.deb -O moby-runc.deb

wget https://packages.microsoft.com/ubuntu/18.04/prod/pool/main/m/moby-containerd/moby-containerd\_1.5.11+azure-1\_amd64.deb -O containerd.io.deb

wget https://packages.microsoft.com/ubuntu/18.04/prod/pool/main/m/moby-cli/moby-cli\_20.10.16+azure-3\_amd64.deb -O moby-cli.deb

wget https://packages.microsoft.com/ubuntu/18.04/prod/pool/main/m/moby-engine/moby-engine\_20.10.16+azure-2\_amd64.deb -O moby-engine.deb

1. Download the IoT Edge Identity Service

curl -L https://github.com/Azure/azure-iotedge/releases/download/1.2.8/aziot-identity-service\_1.2.6-1\_ubuntu18.04\_amd64.deb -o aziot-identity-service.deb

1. Download the IoT Edge runtime

curl -L https://github.com/Azure/azure-iotedge/releases/download/1.2.8/aziot-edge\_1.2.8-1\_ubuntu18.04\_amd64.deb -o aziot-edge.deb

1. Copy all the package files downloaded in steps 1 through 4 to the new IoT Edge device

**Gather the container images**

The following set of steps need to be executed on a device with internet access and Docker installed.

1. Make sure Docker is installed on your local machine (the machine where you downloaded all of the packages previously) and run the following commands to pull the Azure IoT Edge runtime container images to your local repository

sudo docker pull mcr.microsoft.com/azureiotedge-diagnostics:1.2.8

sudo docker pull mcr.microsoft.com/azureiotedge-agent:1.2

sudo docker pull mcr.microsoft.com/azureiotedge-hub:1.2

1. Use the docker save command to download the following container images from mcr.microsoft.com:

sudo docker save mcr.microsoft.com/azureiotedge-diagnostics:1.2.8 | gzip > azureiotedge-diagnostics\_1.2.8.tar.gz

sudo docker save mcr.microsoft.com/azureiotedge-agent:1.2 | gzip > azureiotedge-agent\_1.2.tar.gz

sudo docker save mcr.microsoft.com/azureiotedge-hub:1.2 | gzip > azureiotedge-hub\_1.2.tar.gz

1. Copy all the tar files created in steps 6 and 7 to the new IoT Edge device

**Register the new Azure IoT Edge device**

1. Use the Azure Portal, Azure CLI, or Azure PowerShell to register the newly created IoT Edge device with the IoT Hub. Choose Symmetric Key as the authentication method and copy the value of the primary key once the device registration has been created.

**Install Azure IoT Edge on the device**

The following steps need to be executed on the newly created Azure IoT Edge device from the folder where you copied the package files and container images.

1. Add Microsoft package signing keys to your list of trusted keys

sudo dpkg -i packages-microsoft-prod.deb

1. Install the container engine
   1. Docker

sudo dpkg -i containerd.io.deb

sudo dpkg -i docker-ce-cli.deb

sudo dpkg -i package.deb

* 1. Moby

sudo dpkg -i moby-runc.deb

sudo dpkg -i containerd.io.deb

sudo dpkg -i moby-cli.deb

sudo dpkg -i moby-engine.deb

1. Update the Docker log file policy in /etc/docker/daemon.json as follows (if the files does not exist, create it):

{

"log-driver": "json-file",

"log-opts": {

"max-size": "10m",

"max-file": "3"

}

}

1. Restart Docker / Moby (sudo systemctl restart docker)
2. Use docker load to upload the downloaded container images to the local repository

sudo docker load < azureiotedge-diagnostics\_1.2.8.tar.gz

sudo docker load < azureiotedge-agent\_1.2.tar.gz

sudo docker load < azureiotedge-hub\_1.2.tar.gz

1. If this IoT Edge device will connect to the IoT Hub through an IoT Edge transparent gateway use the docker tag command to tag the uploaded images with the appropriate repository name and version so they are recognized by the IoT Edge runtime

sudo docker tag mcr.microsoft.com/azureiotedge-diagnostics:1.2.8 <ip address or FQDN of the gateway device>/azureiotedge-diagnostics:1.2.8

sudo docker tag mcr.microsoft.com/azureiotedge-agent:1.2 <ip address or FQDN of the gateway device>:443/azureiotedge-agent:1.2

sudo docker tag mcr.microsoft.com/azureiotedge-hub:1.2 <ip address or FQDN of the gateway device>:443/azureiotedge-hub:1.2

1. Install the IoT Edge Identity Service

sudo apt-get install ./aziot-identity-service.deb

1. Install the IoT Edge runtime

sudo apt-get install ./aziot-edge.deb

1. Copy the install.sh, config.toml, and iotedge\_config\_cli\_root.pem files created by the iotedge-config tool to the newly created IoT Edge device.
2. Use nano to edit the config.toml file.
   1. Set the hostname to the FQDN or IP address of the newly created IoT Edge device
   2. If this device will connect directly to the IoT Hub, comment out the parent\_hostname entry
   3. If this device will connect to the IoT Hub using an IoT Edge transparent gateway, set the parent\_hostname to the FQDN or IP address of the gateway device
   4. Set the device\_id to the device ID you used to register the device with the IoT Hub
   5. Set the iothub\_hostname to the Hostname of the IoT Hub
   6. Set the value of the [provisioning.authentication.device\_id\_pk] to the primary key that was copied from the device registration in step 9.
3. Use nano to edit the install.sh file.
   1. Set the device\_id to the device ID you used to register the device with the IoT Hub
4. Execute the install.sh script to copy the certificate(s) to the appropriate folder, copy the config.toml file to the appropriate folder, and configure the IoT Edge runtime.
5. Wait a few moments and the run sudo iotedge check.