Document information

<table>
<thead>
<tr>
<th>Information</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keywords</td>
<td>EdgeLock SE05x, Azure IoT Hub, Secure cloud onboarding</td>
</tr>
<tr>
<td>Abstract</td>
<td>This application note describes how to leverage the EdgeLock SE05x ease of use configuration for secure cloud onboarding to the Azure IoT Hub cloud platform. It provides detailed instructions to run the software example provided as part of the support package using an OM-SE050ARD and an iMX6UltraLite board with a Linux OS.</td>
</tr>
</tbody>
</table>
# Revision history

<table>
<thead>
<tr>
<th>Revision number</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>2019-06-08</td>
<td>First document release</td>
</tr>
<tr>
<td>1.1</td>
<td>2019-06-21</td>
<td>Added correct reference to AN12396</td>
</tr>
<tr>
<td>2.0</td>
<td>2019-11-28</td>
<td>Added EdgeLock SE050 ease of use configuration based on i.MX6UltraLite</td>
</tr>
<tr>
<td>2.1</td>
<td>2020-12-01</td>
<td>Added Section 4. Registering a CA certificate for device onboarding.</td>
</tr>
<tr>
<td>2.2</td>
<td>2020-12-07</td>
<td>Updated to latest template and fixed broken URLs</td>
</tr>
</tbody>
</table>
1 EdgeLock SE05x ease of use configuration

The IoT device identity should be unique, verifiable and trustworthy so that device registration attempts and any data uploaded to Azure IoT Hub can be trusted by the OEM. Azure IoT Hub verifies the device identity using PKI cryptography. This authentication scheme requires that the associated private key remains secret and hidden from users, software or malicious attackers during the product’s lifecycle.

The EdgeLock SE05x security IC is designed to provide a tamper-resistant platform to safely store keys and credentials needed for device authentication and device onboarding to cloud service platforms such as Azure IoT Hub. Using the EdgeLock SE05x security IC, OEMs can safely connect their devices to Azure IoT Hub without writing security code or exposing credentials or keys.

However, key generation and injection into security ICs can introduce vulnerabilities if not done properly. Manual provisioning can lead to errors and is difficult to scale when more devices are needed. Also, to ensure keys are kept safe, injection should take place in a trusted environment, in a facility with security features like tightly controlled access, careful personnel screening, and secure IT systems that protect against cyberattacks and theft of credentials, among others.

In order to allow OEMs to get rid of the complexity of key management and to offload the cost of ownership of a PKI infrastructure, the EdgeLock SE05x is offered pre-provisioned for ease of use. This means that OEMs are not required to program additional credentials and can leverage the EdgeLock SE05x ease of use configuration for most of the use cases, including for secure cloud onboarding of their devices to Azure IoT Hub.
2 Leveraging EdgeLock SE05x ease of use configuration for Azure IoT Hub

Azure IoT Hub uses X.509 certificate-based attestation mechanisms for confirming the device authenticity during a registration attempt. This authentication scheme requires that the associated private key remains secret and hidden from users, software or malicious attackers during the product's lifecycle.

The EdgeLock SE05x is offered off-the-shelf pre-provisioned so that OEMs are not required to program any additional credentials onboard their devices to Azure IoT Hub. It provides a tamper-resistant memory to safely store keys and credentials needed for device authentication and registration to Azure IoT Hub service. Leveraging the EdgeLock SE05x security IC, OEMs can securely connect their devices to Azure IoT Hub without writing security code or exposing credentials or keys.

*Figure 1* illustrates the device registration flow to Azure IoT Hub leveraging the EdgeLock SE05x ease of use configuration:

1. NXP delivers a quantity of EdgeLock SE05x ICs based on a purchase order to a programming facility.
2. The OEM's device manufacturer assembles the EdgeLock SE05x ICs and deploys the software into the final IoT devices. It also needs to take care to read out the device certificate from the EdgeLock SE05x samples.
3. The OEM, as the system operator, manages the Azure IoT Hub account and registers on it every device by registering its device certificate.
4. IoT devices boot up and automatically connect to Azure IoT Hub service using the pre-provisioned credentials inside EdgeLock SE05x.

*Disclaimer*: The described device registration flow spans multiple roles given the various entities involved. How each role is mapped in the registration flow might be scenario-dependent for each OEM.

*Figure 1*. Azure IoT Hub device registration flow
3 Running Azure IoT Hub device onboarding project example

The Azure IoT Hub project example showcases how to leverage EdgeLock SE05x to set up trusted connections to Azure IoT Hub cloud. This section explains how to run the Azure IoT Hub project example included as part of the EdgeLock SE05x support package using the OM-SE050ARD and i.MX6UltraLite boards.

**Note:** The Azure IoT Hub device onboarding procedure described in this section and the Azure IoT Hub demo example are provided only for evaluation purposes. Therefore, the subsequent procedure must be adapted and adjusted accordingly for a commercial deployment.

3.1 Hardware required

This guide provides detailed instructions to the Azure IoT Hub project example using the hardware described below:

1. OM-SE050ARD development kit:

<table>
<thead>
<tr>
<th>Part number</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>OM-SE050ARD</td>
<td>935383282598 EdgeLock SE050 development board</td>
</tr>
</tbody>
</table>

   **Table 1. OM-SE050ARD development kit details**

   i.MX6Ultralite board

<table>
<thead>
<tr>
<th>Part number</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCIMX6UL-EVKB</td>
<td>935328353598 i.MX6UltraLite evaluation kit</td>
</tr>
</tbody>
</table>

   **Table 2. i.MX6Ultralite**

   **Note:** This guide uses OM-SE050ARD, but OM-SE051ARD can be used as well with the same configuration.

3.2 Download EdgeLock SE05x Plug & Trust middleware SD Card Image

The Azure IoT Hub device onboarding project example is included as part of the EdgeLock SE05x Plug & Trust middleware built in the EdgeLock SE05x Plug & Trust middleware SD Card Image. Prepare your SD card following these steps:

1. Download the EdgeLock SE05x Plug & Trust middleware SD Card Image, publicly available from the [NXP website](https://www.nxp.com).
2. Flash the EdgeLock SE05x Plug & Trust middleware SD Card Image into a blank SD card.

**Note:** Refer to AN12397 - Quick start guide with i.MX6UltraLite for detailed instructions about how to prepare a micro-SD card with the pre-compiled Linux image for i.MX6UltraLite board.

### 3.3 Read out device certificate from EdgeLock SE05x ease of use configuration

Azure IoT Hub requires devices to register their certificate in order to connect to their cloud platform. The device certificate can be directly read from the EdgeLock SE05x ease of use configuration. **Table 3** shows the ECC256 key pair we will use for Azure IoT Hub device onboarding. This ECC256 key pair has been selected as an example, for a complete detail of the EdgeLock SE05x ease of use configuration, refer to AN12436 - SE050 configurations.

<table>
<thead>
<tr>
<th>Key name and type</th>
<th>Certificate</th>
<th>Usage policy</th>
<th>Erasable by customer</th>
<th>Identifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cloud connection key 0, ECC256, Die Individual</td>
<td>Cloud Connectivity Certificate 0, ECC signed</td>
<td>Default</td>
<td>Yes</td>
<td>Key: 0xF0000100 Cert: 0xF0000101</td>
</tr>
</tbody>
</table>

Table 3. ECC256 public key used for Azure IoT Hub device onboarding

Before retrieving the pre-provisioned credentials in EdgeLock SE05x, you need to have an SD card flashed with the pre-compiled Linux image. We will use the ssscli tool included in the EdgeLock SE05x Plug & Trust middleware to communicate with the EdgeLock SE05x and to extract the device certificate.

**Note:** Check AN12397-Quick start guide with i.MX6UltraLite for detailed instruction on:
- How to prepare a micro-SD card with the pre-compiled Linux image for i.MX6UltraLite board.
- How to install the USB to UART Bridge VCOM driver in your laptop.
- How to control i.MX6UltraLite board from the TeraTerm terminal application.
- How to boot the i.MX6UltraLite board.
1. Insert the SD card image into the i.MX6UltraLite board, connect it to power supply, to the laptop and boot it up as shown in Figure 2.

![Figure 2. Boot up i.MX6UltraLite board.](image)

2. Open TeraTerm, go to Setup > Serial Port and configure the terminal to 115200 baud rate, 8 data bits, no parity and 1 stop bit and click OK as shown in Figure 3.

![Figure 3. Configure TeraTerm](image)

3. Go to the `/se050_mw_vXX.XX.XX/simw-top/demos/linux/azure` directory, where `vXX.XX.XX` corresponds to the EdgeLock SE05x Plug & Trust middleware version number. Send `> cd se050_mw_vXX.XX.XX/simw-top/demos/linux/azure`. 
4. Follow these steps to open the `ssscli` tool and retrieve the device certificate and the private key reference. Figure 4 shows the execution of these steps.
   a. Send `> ssscli connect se050 tloic2c 0`.
   b. Send `> ssscli get cert 0xF0000101 ./device_cert.cer`.
   c. Send `> ssscli refpem ecc pair 0xF0000100 ./reference_key.pem`.
   d. Send `> ssscli disconnect`.

![Figure 4. Execution of ssscli commands to retrieve credentials.](image)
5. Once the credentials have been extracted from the EdgeLock SE05x, we need to upload them to our Azure Device Provisioning Service instance. To do so, copy them in a USB device following the steps shown in Figure 5:
   a. Connect a USB device in the available port in the i.MX6UltraLite board.
   b. Mount the device in the /home/root/mnt directory.
      Send > mount /dev/sda1 /home/root/mnt.
   c. Copy the extracted key.
      Send > cp reference_key.pem /home/root/mnt.
   d. Copy the extracted certificate.
      Send > cp device_cert.cer /home/root/mnt.
   e. Unmount the device.
      Send > umount /home/root/mnt.

Figure 5. Execution of ssscli commands to copy credentials in an external drive.
6. Using OpenSSL, we will extract the **Common Name** of the device certificate. This Common Name will be used to enroll the device in Azure platform in Section 3.7. Send:

```bash
openssl x509 -in device_cert.cer -text -noout.
```

Figure 6. Check the contents of the device certificate to extract the Common Name.

### 3.4 Create Azure subscription

Microsoft Azure offers a free 30-day trial period to all new account holders. To create a free account in Azure:
1. Go to https://www.azure.com and click the green Start free button as shown in Figure 7:

![Figure 7. Azure subscription creation](image)

2. If you already have an account with Microsoft Office 365, you will be prompted to log in. When you log in, some of your details may already be there as shown in Figure 8:

![Figure 8. Azure free account sign up](image)
3. Supply a valid credit or debit card in the form shown in Figure 9. Microsoft will use it to verify your identity. There is no charge involved with the setting up of a trial account. However, there might be a record for a $0 transaction on your bank statement.

![Figure 9. Azure sign up verification by card](image)

4. Click *I agree* and click *Sign Up* as shown in Figure 10. In a few seconds, your account will be ready.

![Figure 10. Azure sign up agreement acceptance](image)
5. Your Microsoft Azure account has been created. To continue, click the green Go to the portal button or go straight to the Microsoft Azure Portal as shown in Figure 11:

![Azure subscription completion](image)

**Figure 11. Azure subscription completion**

### 3.5 Create an Azure IoT Hub

This section describes how to create an Azure IoT Hub instance using the Azure portal. The Azure IoT Hub is a central message hub for secure bidirectional communication between the cloud-hosted application and the IoT devices.

To use the steps described in this section, you need an Azure subscription. If you do not have an Azure subscription yet, check Section 3.4 to create a free account. If you already have an Azure subscription:
1. Log in the [Azure portal](https://portal.azure.com/). In the Azure dashboard do as shown in Figure 12:
   a. Choose **Create resource**.
   b. Choose **Internet of Things**
   c. Choose **IoT Hub**

![Figure 12. Create an Azure IoT Hub](image)
2. In the **Basics** tab, fill in the fields as shown in Figure 13:
   a. **Subscription**: The subscription to use for your Azure IoT Hub. Select Free trial to use the free account created in Section 3.4.
   b. **Resource group**: A resource group is a container that holds related resources for an Azure solution. To create a new one, click **Create new** and fill in the name for your group.
   c. **Region**: Select the region in which you want your hub to be located.
   d. **IoT Hub name**: Write the name for your Azure IoT Hub. Note that this name must be globally unique.
   e. Click **Next: Size and scale**:

![Figure 13. Create an Azure IoT Hub - Basics form](image-url)
3. On the **Size and scale** tab, do as shown in **Figure 14**:  
   a. Select **F1: Free tier**  
   b. Click **Review+create** at the bottom  
   The **F1: Free tier** that is meant for testing and evaluation. It has all the capabilities of the standard tier, but limited messaging allowances.

![Figure 14. Create an Azure IoT Hub - Size and scale form](image-url)
4. On the **Review+create** tab, click **Create** to confirm the Azure IoT Hub creation as shown in **Figure 15**. You might need to wait a few minutes until your deployment is ready:

---

**Figure 15.** Create an Azure IoT Hub - Review and create form
5. When the Azure IoT Hub instance is deployed, you can see the deployment details and go to the recently created Azure IoT Hub resource as shown in Figure 16:

![Figure 16. Create an Azure IoT Hub completion](image)

3.6 Create an Azure Device Provisioning Service

This section describes how to set up a Device Provisioning Service using Azure portal. The IoT Hub Device Provisioning Service (DPS) is a service that enables zero-touch, just-in-time provisioning to an IoT Hub instance. Follow these steps:

1. In the Azure Dashboard, do as indicated in Figure 17:
   a. Choose Create a resource.
   b. Choose Internet of Things and click on See all.
   c. Choose IoT Hub Device Provisioning Service.

![Figure 17. Create an IoT Hub Device Provisioning Service](image)
2. Choose **Create** in the window that will open and fill the information of the IoT Hub Device Provisioning Service as shown in **Figure 18**. Then, click the **Create** button.

![Figure 18. Provide the basic information of the IoT Hub Device Provisioning Service](image)

3. Check in the Azure dashboard that the IoT Hub Device Provisioning Service instance has been deployed, as shown in **Figure 19**:

![Figure 19. IoT Hub Device Provisioning Service instance in Azure Dashboard](image)
4. Click in your IoT Hub Device Provisioning Service instance and check the detailed information. Write down the **ID Scope** from the **Overview** tab. It will be necessary in **Section 3.8**.

![IoT Hub Device Provisioning Service overview](image)

**Figure 20.** IoT Hub Device Provisioning Service overview.

### 3.7 Add individual enrollment in Device Provisioning Service

This section describes how to create an enrollment in the Azure IoT Hub Device Provisioning Service. An enrollment creates a record of a single device or a group of devices that may register in an IoT Hub Device Provisioning Service.

1. In the Dashboard, select the IoT Hub Device Provisioning Service instance you created in **Section 3.6**.

![IoT Hub Device Provisioning Service instance in Azure Dashboard](image)

**Figure 21.** IoT Hub Device Provisioning Service instance in Azure Dashboard.
2. Choose **Manage Enrollments** from the Azure IoT Hub Device Provisioning Service dashboard as shown in Figure 22.

3. Click on the **Add individual enrollment** option.

![Figure 22. Overview of the IoT Hub Device Provisioning Service instance.](image)

![Figure 23. Manage enrollments.](image)
4. Follow these steps to fill in the enrollment information:
   a. Upload the device certificate (device_cert.cer) extracted in Section 3.3 in the **Primary Certificate** field.
   b. In the **IoT Hub Device ID** field, type the Common Name extracted in Section 3.3.
   c. Set **IoT Edge device** to True.
   d. In the **Select the IoT hubs this device can be assigned to** field, link the IoT Hub created in Section 3.5.
   e. Click **Save** to finish the configuration.

3.8 Register the device to Azure IoT Hub

Follow these steps to register the device certificate to Azure IoT Hub Device Provisioning Service:
1. Connect the board to Internet using an Ethernet cable and boot up the i.MX6UltraLite board as shown in Figure 25:

![Figure 25. Connect board to Internet](image)

2. Export the `OPENSSL_CONF` environment variable as shown in Figure 26:
   a. Save the variable value using the `export` command.
      ```
      Send > export OPENSSL_CONF=/home/root/se050_mw_v02.11.01/simw-top/demos/linux/common/openssl_sss_se050.cnf
      ```
   b. Check that the variable has been correctly exported.
      ```
      Send > printenv
      ```
3. Run the `azure_imx_register` script. The input parameters of the `azure_imx_register` script are listed below:
   • `--registerid` is the registration ID of the device, i.e. the Common Name of the device certificate, extracted in Section 3.3.
   • `--idscope` is the idScope parameter from the IoT Hub Device Provisioning Service, extracted in Section 3.6.
   • `--devcert` is the path to the device certificate, extracted in Section 3.3.
   • `--keypath` is the path to the reference key .pem file, reference in Section 3.3.
   • `--rootpath` is the path to the Azure Root CA certificate, already included in simw-top/demos/linux/azure EdgeLock SE05x Plug & Trust middleware folder.

   Send > ./azure_imx_register --registerid CloudConn0-ECC-04005001C02EDCE5C8C39E0425055955000 --idscope 0ne0008A4E7
--devcert device_cert.cer --keypath reference_key.pem --rootpath azureRootCA.pem.

Figure 27 shows the output of running the azure_imx_register script.

Figure 27. Run the azure_imx_register script.
4. The device registration can be verified in the *Manage enrollments* section of our IoT Hub Device Provisioning Service. The **Status** of the enrollment should be **Assigned** as shown in **Figure 28**.
5. If you run the `ls` command, you will observe a new `CloudConn.txt` file in the directory as shown in Figure 29. This file will be used in Section 3.9 for the connection of the device.

![Figure 29. CloudConn.txt file creation.](image)

This file contains information related to the device onboarding to Azure IoT Hub. Figure 30 shows the contents of the `CloudConn.txt` file.

![Figure 30. CloudConn.txt file contents](image)

### 3.9 Run Azure IoT Hub project example

After device registration in Azure IoT Hub, we can connect the device to the Azure IoT Hub project example. Follow these steps:
1. Execute the `azure_imx_connect` script.
   Send > ./azure_imx_connect --json CloudConn0-ECC-04005001C02EDCE5C8C39E0425055950000.txt.

Figure 31. Output of the `azure_imx_connect` script.
2. To confirm the device connection, go to IoT Hub in the Azure dashboard and click on **IoT Edge** as shown in Figure 32.
3. The device should appear in the list of IoT Edge devices, as shown in Figure 33.

Figure 33. List of connected IoT Edge devices.

In addition, you can also check your device connection in the Azure IoT Hub monitoring metrics. Select **Telemetry messages** under **Metric** menu as shown in Figure 34, a peak of activity can be observed after `azure_imx_connect` is executed.

Figure 34. Telemetry activity reflected in the Azure IoT Hub.
4 Registering a CA certificate for device onboarding

Alternatively to the procedure explained in Section 3, you can configure a CA certificate to enable device certificates it has signed to register with Azure IoT Hub automatically the first time the device connects to Azure IoT Hub. To register device certificates when a client connects to Azure IoT Hub for the first time, you must enable the CA certificate for automatic registration and configure the first connection by the device to provide the required certificates.

This section generates an injects your own credentials in EdgeLock SE05x using the provisioning scripts included as part of EdgeLock SE05x Plug & Trust middleware. Please use this procedure only if you prefer to generate your own keys instead of leveraging the EdgeLock SE05x ease of use configuration used in Section 3.

**Note:** The key generation and injection procedure described in this section is only applicable for evaluation or testing purposes. In a commercial deployment, key provisioning must take place in a trusted environment, in a facility with security features such as tightly controlled access, careful personnel screening, and secure IT systems that protect against cyberattacks and theft of credentials.

4.1 Hardware required

This section provides detailed instructions to the Azure IoT Hub project example using the hardware described below. However, you could use other MCU boards supported by EdgeLock SE05x Plug & Trust middleware for this purpose as well.

1. OM-SE050ARD development kit:

<table>
<thead>
<tr>
<th>Part number</th>
<th>12NC</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>OM-SE050ARD</td>
<td>935383282598</td>
<td>EdgeLock SE050 development board</td>
</tr>
</tbody>
</table>

2. FRDM-K64F board:

<table>
<thead>
<tr>
<th>Part number</th>
<th>12NC</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>FRDM-64F</td>
<td>935326293598</td>
<td>Freedom development platform for Kinetis K64, K63 and K24 MCUs</td>
</tr>
</tbody>
</table>

**Note:** This guide uses OM-SE050ARD, but OM-SE051ARD can be used as well with the same configuration.
4.2 Running Azure IoT Hub key provisioning scripts

This section explains how to generate the credentials for the EdgeLock SE05x using the key provisioning script included in EdgeLock SE05x Plug & Trust middleware and a FRDM-K64F board as a host platform. These credentials are required for the device onboarding into Azure IoT Hub.

Note: Check AN12396- Quick start guide to Kinetis K64 for detailed instructions on how to bring up the FRDM-K64F board.

4.2.1 Download EdgeLock SE05x Plug & Trust middleware

The EdgeLock SE05x Plug & Trust middleware stack includes several MCUXpresso project examples for cloud service onboarding. To download EdgeLock SE05x Plug & Trust middleware:

1. Download EdgeLock SE05x Plug & Trust middleware from www.nxp.com/se050.
2. Create a folder called se050_middleware in C: directory as shown in Figure 35:

![Figure 35. Create se050_middleware folder](image)
3. Create a folder called `simw-top` inside the `se050_middleware`. Unzip `SE050` middleware inside the `simw-top` folder. The unzipped package should look as shown in Figure 36.

![Figure 36. Unzip se050 middleware](image)

**Note:** It is recommended to keep `simw-top` with the **shortest** path possible and **without spaces** in it. This avoids some issues that could appear when building the middleware if the path contains spaces.

4.2.2 Flash FRDM-K64F with VCOM software

The VCOM software allows the FRDM-K64F board to be used as a bridge between the Windows machine and the EdgeLock SE05x and enables the execution of the EdgeLock SE05x `ssscli` tool and other utilities from the laptop. To flash the VCOM software into the FRDM-K64F, follow these steps:

1. Unplug and plug again the USB cable to the openSDA USB port as shown in Figure 37:

![Figure 37. Unplug and plug OpenSDA port](image)
2. When you plug the board, your laptop should recognize the board as an external drive as shown in Figure 38:

![Figure 38. FRDM-K64F drive](image)

3. Flash the VCOM software to FRDM-K64F. The VCOM software binary can be found in the EdgeLock SE05x Plug & Trust middleware package, inside the `simw-top\binaries` folder as shown in Figure 39:

![Figure 39. VCOM binary folder](image)
4. Drag and drop or copy and paste the a7x_vcom-T1oI2C-frdmk64f-SE050x.bin file into the FRDM-K64F drive from your computer file explorer as shown in Figure 40:

![Figure 40. Drag and drop VCOM binary](image)

5. The serial and VCOM ports should be recognized by your Device Manager. To check that the ports are recognized, follow the steps indicated in Figure 41:
   a. Unplug the USB cable from the OpenSDA USB port.
   b. Plug the USB cable to the OpenSDA USB port.
   c. Check that the serial port is recognized in the category **Ports (COM & LTP)**. In this document, it is recognized as **USB Serial Device (COM7)** but this naming might change depending on your computer. Therefore, it is important that you identify which device is recognized at the moment you plug the SDA USB port to the computer.
   d. Plug the USB cable to the K64F USB port.
   e. Check that the VCOM port is recognized in the category **Ports (COM & LTP)**. In this document, it is recognized as **Virtual Com Port (COM8)** but this naming might change depending on your computer (e.g. It could also appear named as
USB Serial Device). Therefore, it is important that you identify which device is recognized at the moment you plug the K64F USB port to the computer.

![Check VCOM and serial ports](image)

**Figure 41. Check VCOM and serial ports**

*Note: Please note that it is possible that either of the two COM ports is not detected when using low-quality or charge-only USB cables.*

### 4.2.3 Key and certificate configuration for use with Azure IoT Hub

The EdgeLock SE05x Plug & Trust middleware includes an executable file that allows you to easily generate some sample credentials and inject them into the EdgeLock SE05x for their use with this Azure IoT Hub demo.

To externally generate the keys and inject them into the EdgeLock SE05x, follow these steps:

1. Mount OM-SE050ARD on top of the FRDM-K64F. Then, connect FRDM-K64F OpenSDA port and K64F port to your laptop as shown in Figure 42

![Connect boards](image)

**Figure 42. Connect boards**
2. Go to `simw-top\binaries\pySSSCLI` folder and locate the `Provision_AZURE.exe` file as shown in Figure 43.

![Figure 43. Find Provision_AZURE.exe file in your EdgeLock SE05x Plug & Trust middleware package](image)

3. Open a command prompt
4. Use the `Provision_AZURE.exe` executable to generate and inject keys into your EdgeLock SE05x. Go to the folder `simw-top\binaries\pySSCLI` and follow the steps shown in Figure 44:
   a. Run the executable `Provision_AZURE.exe <K64_COM_port_number>`. For that, you also need to indicate the VCOM port number corresponding to the K64 USB port of your board (See here).
   Send `>Provision_AZURE.exe COM8`
   b. Check that the keys are generated and injected.
   c. Check that the program execution completes successfully.

![Figure 44. Run Provision_AZURE.exe executable](image-url)
5. Go to \simw-top\binaries\pySSCLI\azure folder and check that the keys appear inside the folder as shown in Figure 45:

![Figure 45. Generated Azure credentials](image)

### 4.3 Azure IoT Hub account setup

This section details the steps related to the Azure IoT Hub account configuration.

#### 4.3.1 Create Azure subscription

Create an Azure subscription following the steps described in Section 3.4.

#### 4.3.2 Create an Azure IoT Hub

Create an Azure IoT Hub instance following the steps described in Section 3.5.

#### 4.3.3 Create a device

This section describes how to create a device instance using the Azure IoT Hub portal. Follow these steps:
1. In your Azure IoT Hub resource, do as shown in Figure 46:
   a. Go to IoT devices on the left hand side menu.
   b. Click New to create a new device.

Figure 46. Create a device
2. To create a device, follow the indications in
   a. As device ID, copy the 24 digits generated in the certificate credential in Figure 47
   b. Select **X.509 CA Signed** as authentication type.
   c. Click **Save**.

![Figure 47. Create a device II](image_url)

### 4.3.4 Upload root CA certificate to your Azure IoT Hub

Azure IoT Hub supports three attestation mechanisms for confirming the device authenticity during registration based on:

- **X.509 certificates**
- **Trusted Platform Module (TPM)**
- **Symmetric key**

This document describes how to use X.509 CA certificates to authenticate devices connecting to the Azure IoT Hub. The other two attestation mechanisms are beyond the scope of this application note. To upload a new root CA certificate:
1. In your Azure IoT Hub resource, click in **Certificates** menu (1) and then click **Add** (2) as shown in **Figure 48**:

![Figure 48. Upload a root CA certificate](image1.png)

Then, type the certificate name and select the certificate to upload, as shown in:

- **As Certificate Name**, for instance, use `rootCA_QMC`.
- **Upload the rootCA certificate that we generated in **Figure 49**.**

![Figure 49. Upload a root CA certificate II](image2.png)
2. The certificate we have just uploaded will appear in the list as **Unverified** as shown in Figure 50.

The certificate status remains in **Unverified** until we complete the **proof of possession** validation. The **proof of possession** mechanism verifies that the uploader is in possession of the certificate private key. For this verification, the Azure IoT Hub generates a **verification code** that needs to be included in a **verification certificate** signed by the CA certificate private key.

![Figure 50. Root CA certificate unverified status](image)
3. To obtain the verification code, do as shown in Figure 51
   a. Click on the certificate uploaded in the previous step
   b. Click **Generate Verification code**
   c. Copy and save the generated verification code. This code will be used to conduct the proof of verification and certificate ownership.

![Figure 51. Obtain root CA certificate verification code](image-url)
4. Go to `simw-top\pycli\Provisioning` folder and execute the `verification_certificate.py` script, which require three arguments:
   `<RootCA_certificate>`
   `<RootCA_private key>`
   `<Azure_verification_code>`
   The first two correspond to credentials injected and the third, is the verification code we have just obtained from Azure. Figure 52 is an example of the command to be sent:

   ![](image)

   **Figure 52. Generate verification certificate**

5. Make sure a file called `verifyCert.cer` has been generated in `simw-top\pycli\Provisioning` folder, as shown in Figure 53:

   ![](image)

   **Figure 53. Verification certificate generated**

### 4.3.5 Verify root CA certificate

To change the root CA certificate status from `Unverified` to `Verified`:
1. In the Azure IoT Hub portal, do as shown in Figure 54:
   a. Click on verification certificate .pem or .cer file button
   b. Upload verifyCert4.cer, which is the signed verification certificate generated in Section 4.3.4
   c. Click Open
   d. Click Verify

   ![Figure 54. Root CA certificate verification](image)

2. Check that the proof of possession verification was successful. The certificate should have changed its status to Verified as shown in Figure 55. The registration of the OEM's root CA certificate is a one-time process.

   ![Figure 55. Root CA certificate verified](image)
4.4 Azure IoT Hub project configuration

To run the Azure IoT Hub project example using the FRDM-K64F board, we need to:

- Download and install the FRDM-K64F SDK
- Import Azure IoT Hub example project
- Configure Azure IoT Hub project account settings

**Note:** Before running the Azure IoT Hub demo example, you need to have installed MCUXpresso IDE and FRDM-K64F SDK in your local environment and imported the Azure IoT Hub project example. Check AN12396- Quick start guide to Kinetis K64 for detailed instructions on:

- How to install MCUXpresso
- How to obtain FRDM-K64F SDK
- How to import FRDM-K64F project examples, including Azure IoT Hub project example.

4.4.1 Download and install the FRDM-K64F SDK

The Azure IoT Hub device onboarding project example is included as part of the FRDM-K64F SDK. Install it to your MCUXpresso workspace as shown in Figure 56:

1. Download the FRDM-K64F SDK, publicly available from the NXP website.
2. Drag and drop the FRDM-K64F SDK zip file in the Installed SDKs section in the bottom part of the MCUXpresso IDE.
3. Check that the FRDM-K64F SDK is installed successfully.

![Import FRDM-K64F SDK](image)

**Note:** For more detailed instructions on how to install the FRDM-K64F SDK into our MCUXpresso workspace, refer to AN12396 - Quick start guide with FRDM-K64F.
4.4.2 Import Azure IoT Hub example project

The FRDM-K64F SDK includes a project example called se_SE050x_cloud_azure. Import it to your MCUXpresso workspace as shown in Figure 57:

1. Click Import SDK examples from the MCUXpresso IDE quick start panel.
2. Select se_SE050x_cloud_azure project example and click the Finish button.
3. Check that the project is now visible in your MCUXpresso workspace.

*Note: For detailed instructions on how to import project examples from FRDM-K64F SDK, check AN12396 - Quick start guide with Kinetis K64F.*

![Figure 57. Import Azure project in the workspace](image)

4.4.3 Configure Azure IoT Hub project account settings

We need to change two variables in the MCUXpresso project related with your Azure IoT Hub account settings. In the MCUXpresso workspace:
1. Go to the project explorer and open the `azure_iot_config.h`, which can be found in `source\azure_task\` folder as shown in Figure 58:

![Figure 58. Find `azure_iot_config.h` file](image)

2. Modify the `#define AZURE_IOT_HUB_NAME` macro definition to add your Azure IoT Hub name created in Section 3.5 as shown Figure 59:

![Figure 59. Change `#define AZURE_IOT_HUB_NAME`](image)
3. Modify the `#define AZURE_DEVICE_NAME` macro definition to add your device ID created in Section 4.3.3 as shown in Figure 59:

![Figure 60. Change #define AZURE_DEVICE_NAME](image)

Everything is now fully ready to run the demo on the FRDM-K64F board.

4.5 Azure IoT Hub project execution

Now we are fully ready to run the project on the FRDM-K64F. To start the Azure IoT Hub project example, follow these steps:

1. Connect FRDM-K64F OpenSDA port, K64F port and Ethernet interface to your laptop as shown in Figure 61:

![Figure 61. Connect FRDM-K64F board](image)
2. Open TeraTerm, go to Setup > Serial Port and choose the one corresponding to the OpenSDA port of the board, 115200 baud rate, 8 data bits, no parity and 1 stop bit and click OK as shown below.

![Configure TeraTerm](image)

**Figure 62. Configure TeraTerm**

3. Go to the MCUXpresso Quickstart Panel and click **Debug** button, wait a few seconds until the project executes and click on **Resume** to allow the software to continue its execution as shown in **Figure 63**:

![Debug Azure IoT Hub project](image)

**Figure 63. Debug Azure IoT Hub project**
4. Your device should now be connected to Azure IoT Hub. Check that your device is connected by:
   a. Checking the TeraTerm logs as shown in Figure 64.
   
   ![Figure 64. Device connection to Azure IoT Hub](image)

   b. In addition, you can also check that some activity is traced in your Azure IoT Hub account dashboard, as shown in Figure 65.

   ![Figure 65. Device connection to Azure IoT Hub - dashboard](image)
5  Legal information

5.1 Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

5.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors. In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or work in progress) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory. Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer’s own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification. Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer’s sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer’s applications and products as planned. NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer’s applications or products, or the application or use by customer’s third party customer(s). Customer is responsible for doing all necessary testing for the customer’s applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer’s third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Evaluation products — This product is provided on an “as is” and “with all faults” basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer. In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out of the use or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages. Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer’s exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars ($5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

Translations — A non-English (translated) version of a document is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified or documented vulnerabilities. Customer is responsible for the design and operation of its applications and products through their lifecycles to reduce the extent of these vulnerabilities on customer’s applications and products. Customer’s responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer’s applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately. Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP. NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

5.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are the property of their respective owners.
## Tables

<table>
<thead>
<tr>
<th>Tab.</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>OM-SE050ARD development kit details</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>i.MX6Ultralite</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>ECC256 public key used for Azure IoT Hub device onboarding</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>OM-SE050ARD development kit details</td>
<td>31</td>
</tr>
<tr>
<td>5</td>
<td>FRDM-K64F details</td>
<td>31</td>
</tr>
</tbody>
</table>
Figures

Fig. 1. Azure IoT Hub device registration flow ..........4
Fig. 2. Boot up i.MX6UltraLite board. .................7
Fig. 3. Configure TeraTerm ................................7
Fig. 4. Execution of ssocii commands to retrieve credentials. .............................................8
Fig. 5. Execution of ssocii commands to copy credentials in an external drive .........................9
Fig. 6. Check the contents of the device certificate to extract the Common Name. ..................10
Fig. 7. Azure subscription creation .....................11
Fig. 8. Azure free account sign up .....................11
Fig. 9. Azure sign up verification by card .............12
Fig. 10. Azure sign up agreement acceptance ..........12
Fig. 11. Azure subscription completion ...............13
Fig. 12. Create an Azure IoT Hub ......................14
Fig. 13. Create an Azure IoT Hub - Basics form ......15
Fig. 14. Create an Azure IoT Hub - Size and scale form ......................................................16
Fig. 15. Create an Azure IoT Hub - Review and create form .................................................17
Fig. 16. Create an Azure IoT Hub completion ...........18
Fig. 17. Create an IoT Hub Device Provisioning Service ......................................................18
Fig. 18. Provide the basic information of the IoT Hub Device Provisioning Service .................19
Fig. 19. IoT Hub Device Provisioning Service instance in Azure Dashboard ..........................19
Fig. 20. IoT Hub Device Provisioning Service overview. ..........................................................20
Fig. 21. IoT Hub Device Provisioning Service instance in Azure Dashboard ..........................20
Fig. 22. Overview of the IoT Hub Device Provisioning Service instance. ................................21
Fig. 23. Manage enrollments. .............................21
Fig. 24. Add individual enrollment detail. ..........22
Fig. 25. Connect board to Internet .....................23
Fig. 26. Export the OPENSSL_CONF variable. ....23
Fig. 27. Run the azure_inx_register script ..........25
Fig. 28. Device registered in Azure platform. ....26
Fig. 29. CloudConn .txt file creation. .................27
Fig. 30. CloudConn .txt file contents .....................27
Fig. 31. Output of the azure_inx_connect script ..........28
Fig. 32. Select IoT Edge in IoT Hub overview ........29
Fig. 33. List of connected IoT Edge devices ..........30
Fig. 34. Telemetry activity reflected in the Azure IoT Hub ........................................................30
Fig. 35. Create se050_middleware folder ...............32
Fig. 36. Unzip se050 middleware .......................33
Fig. 37. Unplug and plug OpenSDA port ..........33
Fig. 38. FRDM-K64F drive ................................34
Fig. 39. VCOM binary folder .............................34
Fig. 40. Drag and drop VCOM binary .................35
Fig. 41. Check VCOM and serial ports ...............36
Fig. 42. Connect boards ....................................36
Fig. 43. Find Provision_AZURE.exe file in your EdgeLock SE05x Plug & Trust middleware package .................................................................37
Fig. 44. Run Provision_AZURE.exe executable ........38
Fig. 45. Generated Azure credentials ..................39
Fig. 46. Create a device .................................40
Fig. 47. Create a device II .............................41
Fig. 48. Upload a root CA certificate .................42
Fig. 49. Upload a root CA certificate II .............42
Fig. 50. Root CA certificate unverified status .........43
Fig. 51. Obtain root CA certificate verification code 44
Fig. 52. Generate verification certificate .............45
Fig. 53. Verification certificate generated ..........45
Fig. 54. Root CA certificate verification .............46
Fig. 55. Root CA certificate verified ..................46
Fig. 56. Import FRDM-K64F SDK .......................47
Fig. 57. Import Azure project in the workspace ....48
Fig. 58. Find azure_iot_config.h file ..................49
Fig. 59. Change #define AZURE_IOT_HUB_NAME ....49
Fig. 60. Change #define AZURE_DEVICE_NAME ......50
Fig. 61. Connect FRDM-K64F board .................50
Fig. 62. Configure TeraTerm ............................51
Fig. 63. Debug Azure IoT Hub project ...............51
Fig. 64. Device connection to Azure IoT Hub ..........52
Fig. 65. Device connection to Azure IoT Hub - dashboard .........................52
Contents

1 EdgeLock SE05x ease of use configuration ....................................................... 3
2 Leveraging EdgeLock SE05x ease of use configuration for Azure IoT Hub ..................... 4
3 Running Azure IoT Hub device onboarding project example ........................................... 5
  3.1 Hardware required ............................................. 5
  3.2 Download EdgeLock SE05x Plug & Trust middleware SD Card Image .............................. 5
  3.3 Read out device certificate from EdgeLock SE05x ease of use configuration ....................... 6
  3.4 Create Azure subscription ........................................................................ 10
  3.5 Create an Azure IoT Hub .................................................................... 13
  3.6 Create an Azure Device Provisioning Service ................................................... 18
  3.7 Add individual enrollment in Device Provisioning Service .................................. 20
  3.8 Register the device to Azure IoT Hub ................................................. 22
  3.9 Run Azure IoT Hub project example ................................................. 27
4 Registering a CA certificate for device onboarding ......................................................... 31
  4.1 Hardware required ........................................... 31
  4.2 Running Azure IoT Hub key provisioning scripts .................................................. 32
    4.2.1 Download EdgeLock SE05x Plug & Trust middleware ....................................... 32
    4.2.2 Flash FRDM-K64F with VCOM software ......................................................... 33
    4.2.3 Key and certificate configuration for use with Azure IoT Hub ................................. 36
  4.3 Azure IoT Hub account setup .................................................................. 39
    4.3.1 Create Azure subscription ................................................................. 39
    4.3.2 Create an Azure IoT Hub ................................................................. 39
    4.3.3 Create a device ........................................................................... 39
    4.3.4 Upload root CA certificate to your Azure IoT Hub ..................................... 41
    4.3.5 Verify root CA certificate .................................................................. 45
  4.4 Azure IoT Hub project configuration ....................................................... 47
    4.4.1 Download and install the FRDM-K64F SDK .............................................. 47
    4.4.2 Import Azure IoT Hub example project ........................................... 48
    4.4.3 Configure Azure IoT Hub project account settings .................................. 48
  4.5 Azure IoT Hub project execution ......................................................... 50
5 Legal information .................................................................................. 53