Document information

<table>
<thead>
<tr>
<th>Information</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keywords</td>
<td>SE050, Azure IoT Hub, Secure cloud onboarding</td>
</tr>
<tr>
<td>Abstract</td>
<td>This application note describes how to run the Azure IoT Hub software example included in the SE050 support package. This software example demonstrates how to leverage SE050 security IC for secure cloud onboarding.</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>
</tbody>
</table>
### Abbreviations

Table 1. Abbreviations

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OEM</td>
<td>Original Equipment Manufacturer</td>
</tr>
</tbody>
</table>
1 Read this first

This section lists the hardware and software required before starting this document.

1.1 Prerequisites

- The AN12396- Quick start guide to Kinetis K64 is a prerequisite before running the Azure IoT Hub demo explained in this document.

1.2 Required hardware

The following hardware will be used throughout the document:

1. OM-SE050ARD development kit:

<table>
<thead>
<tr>
<th>Table 2. OM-SE050ARD development kit details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part number</td>
</tr>
<tr>
<td>OM-SE050ARD</td>
</tr>
</tbody>
</table>

2. FRDM-K64F board:

<table>
<thead>
<tr>
<th>Table 3. FRDM-K64F details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part number</td>
</tr>
<tr>
<td>FRDM-64F</td>
</tr>
</tbody>
</table>

1.3 Required software

- Windows 10 machine
2 Azure IoT Hub device registration flow

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 uses X.509 certificate-based attestation mechanisms for confirming the device authenticity during a registration attempt. This authentication scheme requires a certificate chain of trust, from the OEM's CA certificate to the OEM's device certificate as well as their associated private key.

The SE050 is designed to provide a tamper-resistant memory to securely store keys and credentials needed for device authentication and registration to Azure IoT Hub service. Leveraging the SE050 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 using SE050 security IC as a root of trust. It consists of the following steps:

1. NXP delivers a quantity of SE050 ICs based on a purchase order to a programming facility.
2. The SE050 ICs are securely provisioned with die-individual credentials that will be used to register to GCP. The provisioned SE050 samples are delivered to the OEM.
3. The OEM sends the SE050 ICs to the OEM's device manufacturer.
4. The OEM, as the system operator, manages the Azure IoT Hub account and registers the OEM CA certificate.
5. The OEM's device manufacturer assembles the SE050 ICs and deploys the software into the final IoT devices. Manufactured IoT devices are shipped to end customers.
6. IoT devices boot up and automatically connect to Azure IoT Hub service using the pre-provisioned credentials inside SE050.

Figure 1. Azure IoT Hub device registration flow
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.

You can rely on any of the secure provisioning options for the SE050 security IC:

- **SE050 pre-configuration for ease of use**: Every SE050 product variant comes pre-provisioned with root keys which can be used for all major use cases, including secure cloud onboarding.

- **SE050 secure provisioning by NXP**: The NXP Trust Provisioning service offers customized and secure injection of die-individual keys and credentials into SE050 on behalf of the OEM. This service is available for high volume orders of more than 150K units.

- **SE050 secure provisioning by NXP distributors or third-party partners**: NXP has agreements with distributors and third-party partners to offer customized and secure injection of die-individual keys and credentials into SE050 for orders of any size.
3 Azure IoT Hub demo example

The Azure IoT Hub demo is an illustrative software example that showcases how to leverage SE050 to set up trusted connections to Azure IoT Hub cloud. This section explains how to run the Azure IoT Hub demo example included as part of the SE050 support package.

The Azure IoT Hub demo example can be ported to multiple NXP MCUs / MPUs. This section explains the steps required to run the Azure IoT Hub demo for each MCU / MPU platform. You only need to follow the steps related to the MCU / MPU platform you choose for evaluation purposes.

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 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 2:

2. If you already have an account with MicrosoftOffice 365, you will be prompted to log in. When you log in, some of your details may already be there as shown in Figure 3:
3. Supply a valid credit or debit card in the form shown in Figure 4. Microsoft will use 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.
4. Click *I agree* and click *Sign Up* as shown in Figure 5. In a few seconds, your account will be ready.

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 6.
3.2 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 3.1 to create a free account. If you already have an Azure subscription:

1. Log in to the Azure portal. In the Azure dashboard, do as shown in Figure 7:
   a. Choose Create resource.
   b. Choose Internet of Things
   c. Choose IoT Hub

![Figure 6. Azure subscription completion](image_url)

![Figure 7. Create an Azure IoT Hub](image_url)
2. In the Basics tab, fill in the fields as shown in Figure 8:
   a. Subscription: The subscription to use for your Azure IoT Hub. Select Free trial to use the free account created in Section 3.1
   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 8. Create an Azure IoT Hub - Basics form](image)

3. On the Size and scale tab, do as shown in Figure 9:
   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.
4. On the Review+create tab, click Create to confirm the Azure IoT Hub creation as shown in Figure 10. You might need to wait a few minutes until your deployment is ready:
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 11:

Figure 10. Create an Azure IoT Hub - Review and create form

Figure 11. Create an Azure IoT Hub completion
3.3 Create a device

This section describes how to create a device instance using the Azure IoT Hub portal. Follow these steps:

1. Enter to your Azure IoT Hub, doing as indicated in Figure 12:
   a. Go to Dashboard
   b. Click on your Azure IoT Hub resource name

   ![Figure 12. Go to Azure IoT Hub resource](image)

2. In your Azure IoT Hub resource, do as shown in Figure 13:
   a. Go to IoT devices on the left hand side menu.
   b. Click Add to create a new device.

   ![Figure 13. Create a device](image)
3. To create a device, follow the indications in
   a. Choose a device ID (e.g. *mydevice*)
   b. Select **X.509 CA Signed** as authentication type.
   c. Click **Save**.

![Create a device](image)

**Figure 14. Create a device II**

### 3.4 Create your own X.509 certificates

This section explains how to create your own X.509 certificates using a third party tool such as OpenSSL and PowerShell or Bash scripts. The rest of this tutorial uses test CA certificates generated in this section.

- If you are using FRDM-K64F, go to Section 3.4.1 as we will use the PowerShell scripts

**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 like tightly controlled access, careful personnel screening, and secure IT systems that protect against cyberattacks and theft of credentials.

#### 3.4.1 Create your own X.509 certificates using PowerShell

This section explains how to create your own X.509 certificates using OpenSSL and PowerShell scripts.

##### 3.4.1.1 Install OpenSSL

OpenSSL is required prior to running the PowerShell scripts. To get OpenSSL for Windows:

- For places to download: [https://www.openssl.org/docs/faq.html#MISC4](https://www.openssl.org/docs/faq.html#MISC4)
- To build from source: [https://www.openssl.org/source/](https://www.openssl.org/source/)
As an example, you can find Windows binaries for OpenSSL [here](https://www.openssl.org). To install OpenSSL, do the following:

1. Download the latest **Win32 OpenSSL** version as shown [Figure 15](#) (e.g. Win32 OpenSSL v1.1.0.k). This is an OpenSSL for Windows installation package for 32-bit version.

![Figure 15. Download Win32 OpenSSL binary](image)

2. Accept the license agreement as shown in [Figure 16](#):

![Figure 16. OpenSSL license agreement](image)

3. Leave the default destination folder (C:\OpenSSL-Win32) and click **Next** as shown in [Figure 17](#):

![Figure 17](image)
4. Leave the default start menu folder and click *Next* as show in Figure 18:

5. Copy OpenSSL DLLs to the default location (Windows system directory) and click *Next* as shown in Figure 19:
6. Click *Install* to continue with the installation as shown in Figure 20. Wait a few seconds until the installation completes.

7. OpenSSL for Windows has now been installed and can be found in C:\OpenSSL-Win32\ as shown in Figure 21.
3.4.1.2 Set OpenSSL environment variables

To complete the setup, we need to set the OpenSSL environment variables:

1. Open the environment variables menu in your laptop. For that, you can do as shown in Figure 22:
   a. Type environment variables in your Windows browser.
   b. Click on Edit the system environment variables.
   c. On the System Properties window, click on environment variables button.

2. Create the OPENSSL_CONF system variable. Set it to: OPENSSL_CONF=C:\OpenSSL-Win32\bin\openssl.cfg as shown as Figure 23:
3. The `openssl.exe` executable should be located in: `C:\OpenSSL-Win32\bin` directory. Add to the `Path` environment variable the path to the `openssl.exe` executable as shown in Figure 24:
   a. Double click on the `Path` system variable.
   b. Click `New`.
   c. Add the path location to the `openssl.exe;C:\OpenSSL-Win32\bin` and click `OK` button.
4. Check that OpenSSL environment variables are correctly set. For that, you can do:
   a. Open a Command Prompt.
   b. Type `openssl`
   c. An OpenSSL command line should appear as shown in Figure 25. If this is not the case, you might need to revisit your OpenSSL installation or environment variables.

If you have reached this point, OpenSSL is successfully installed in your machine. You can continue with the next steps.

3.4.1.3 Initial setup

The `simw-top\demos\ksdk\azure\scripts` folder contains a `ca-certs.ps1` script to help create test certificates for Azure IoT Hub’s CA Certificate / proof-of-possession.

**Note:** The `ca-certs.ps1` PowerShell script is contained inside the SE050 Plug & Trust middleware. Please, refer to AN12396- Quick start guide to Kinetis K64 for detailed instructions on how to get up and running with the SE050 Plug & Trust middleware.

To create your own X.509 certificates using PowerShell:
1. Start PowerShell as an Administrator as shown in Figure 26:
2. Go to the `simw-top\demos\ksdk\azure\scripts` directory as (Figure 27) > `cd <RootSourceDir>\demos\ksdk\azure\scripts`

3. (Figure 28) Run `Set-ExecutionPolicy -ExecutionPolicy Unrestricted` and type Y

4. (Figure 29) Run `.\ca-certs.ps1` . You might be asked to type R to run the script.
5. (Figure 30) Run Test-CACertsPrerequisites. This script validates whether everything is in place to run the following scripts. If OpenSSL PATH or OPENSSL_CONF environment variables are not defined, this command will return an error. If that is your case, review your OpenSSL installation was done as described in Install OpenSSL.

Note: If you see the error: Test-CACertNotInstalledAlready : Certificate CN=Azure IoT CA TestOnly Root CA already installed in the certificate store, it means that there are already some Azure certificates generated in your machine. You need to remove them from Manage user certificates as shown in Section 3.4.1:
3.4.1.4  Create the certificate chain of trust

To create the certificate chain of trust, follow these steps:

1.  (Figure 32) Run `New-CACertsCertChain ecc`. Note these updates your Windows Certificate storage with these certificates:
Figure 32. PowerShell - New-CaCertsCertChain

2. Check that Intermediate CAs and a Root CA have been created in your folder as shown in Figure 33:

3.4.1.5 Create device certificate

To create the new device certificate:

1. Run `New-CACertsDevice <deviceID>`. The deviceID should be the one created and registered in Azure IoT Hub in Section 3.3. Assume the deviceID is `mydevice`:

```powershell
> New-CACertsDevice mydevice
```
2. Check that files mydevice* that contain the public key, private key, and PFX of this certificate have been created and added in your folder as shown in Figure 35.

Do not close the PowerShell as we will use it later on to generate the verification certificate as explained in Section 3.6.

If you have completed this section, go to Upload root CA certificate to your Azure IoT Hub.

3.5 Upload root CA certificate to your Azure IoT Hub

Azure 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. Enter to your Azure IoT Hub, doing the following:
   a. Go to Dashboard
   b. Click on your Azure IoT Hub resource name
Figure 36. Go to Azure IoT Hub resource

2. In your Azure IoT Hub resource, do as shown in Figure 37:
   a. Open the **Certificates** menu.
   b. Click **Add** to add a new CA certificate.
   c. Enter a display name to your certificate.
   d. Select from your file system the Root CA certificate we created in Section 3.4.
   e. Click **Save**.

Figure 37. Upload a root CA certificate

3. The certificate we have just uploaded will appear in the list as **Unverified** as shown in Figure 38.

   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.
4. To obtain the verification code, do as shown in Figure 39
   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.

5. Continue to Section 3.6 to create your own verification certificate.

3.6 Create verification certificate

This section explains how to create your own Azure IoT Hub verification certificates using a third party tool such as OpenSSL and PowerShell scripts. The rest of this tutorial uses test CA certificates generated in this section.

**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 like tightly controlled access, careful personnel screening, and secure IT systems that protect against cyberattacks and theft of credentials.
3.6.1 Create verification certificate using PowerShell

In Section 3.5, we registered our own root CA with Azure IoT Hub. In this step, we will prove that we actually own it generating a verification certificate. To generate it, do the following:

1. In the PowerShell, run `New-CACertsVerificationCert <verification_code>`. For example, assume Azure IoT Hub verification code was "29946C30CBD81A7EDF7B9EC3A6358FD0ABB1792BE9DA9FFA", the certificate subject name should be that code. See the example below:

   (Figure 40) Run: `New-CACertsVerificationCert 29946C30CBD81A7EDF7B9EC3A6358FD0ABB1792BE9DA9FFA`

   ![PowerShell - NewCaCertsVerificationCert](image)

2. Check that the verification certificate has been created in your folder as shown in Figure 41:

   ![PowerShell - verification certificate](image)

   If you have completed this section, go to Verify root CA certificate
3.7 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 42:
   a. Click on verification certificate .pem or .cer file button
   b. Upload verifyCert4.cer, which is the signed verification certificate generated in Section 3.6
   c. Click Open
   d. Click Verify

   ![Figure 42. 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 43. The registration of the OEM's root CA certificate is a one-time process.

   ![Figure 43. Root CA certificate verified](image)

3.8 Key provisioning

The key provisioning is the process of securely generating and injecting credentials into SE050 security IC. This section explains how to inject credentials generated in Section 3.4 into SE050 using the tools included as part of the SE050 support package.
The tools used to generate and inject keys into SE050 are ported to multiple NXP MCUs / MPUs. This section explains the steps required to generate and inject keys for each MCU / MPU platform. You only need to follow the steps related to the MCU / MPU platform you choose for evaluation purposes.

- If you are using FRDM-K64F, go to Section 3.8.1

**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 like tightly controlled access, careful personnel screening, and secure IT systems that protect against cyberattacks and theft of credentials.

### 3.8.1 Key provisioning using FRDM-K64F

The key provisioning consists of injecting inside the SE050 the device credentials we generated in Section 3.4.

**Note:** Before the key provisioning using FRDM-K64F, you need to have installed the SE050 Plug & Trust middleware in your local environment. Check AN12396- Quick start guide to Kinetis K64 for detailed instructions on how to install the SE050 Plug & Trust middleware in your local environment.

#### 3.8.1.1 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 SE050 and enables the execution of the SE050 `ssscii` 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 44:

   ![Figure 44. 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 45:
3. Flash the VCOM software to FRDM-K64F. The VCOM software binary can be found in the SE050 middleware package, inside the `simw-top\binaries` folder as shown in Figure 46:

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 47:
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 48:
   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 Servial 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.
3.8.1.2 Key injection

To be able to register and connect to Azure IoT Hub, we need to inject into SE050 a key pair and certificate. To do so:

1. Check that files mydevice* that contain the public key, private key, and PFX of this certificate have been created and added in your folder as shown in Figure 49.

2. Start the ssscli tool sending the commands shown in Figure 50:
   a. Go to simw-top\binaries\pySSSCLI folder:
      Send: > cd se050.middleware\simw-top\binaries\pySSSCLI
   b. Check your VCOM port number in your Device Manager. Open the connection:
      Send: > ssscli.exe connect se050 vcom <COM_NUMBER>
   c. Send the reset command:
      Send: > ssscli.exe se05x reset
Figure 50. Start ssscli tool

Note: If you see the following message: WARNING: sss.connect: Session already open, close current session first message as shown in Figure 51, it means that you have a session open. To close it, send: (1) > ssscli disconnect and then send once again (2) > ssscli connect se050 vcom <COM_NUMBER> and later (3) > ssscli se05x reset.

Figure 51. Close an already opened session

3. Inject the private key inside SE050 (mydevice-private.pem): (Figure 52) Send: > ssscli.exe set ecc pair 0x223344 ..\..\demos\ksdk \azure\scripts\mydevice-private.pem

Figure 52. Inject private key

4. Inject the certificate inside SE050 (mydevice-public.pem): (Figure 53) Send: > ssscli.exe set cert 0x223345 ..\..\demos\ksdk \azure\scripts\mydevice-public.pem
5. Close the connection.

(Figure 54) Send: >ssscli.exe disconnect

If you have completed this section, go to Running Azure IoT Hub demo.

3.9 Running Azure IoT Hub demo

This section explains the steps required to run the software example that creates a trusted connection to Azure IoT Hub for each MCU / MPU platform. You only need to follow the steps related to the MCU / MPU platform you choose for evaluation purposes.

- If you are using FRDM-K64F, go to Section 3.9.1

3.9.1 Running Azure IoT Hub demo using FRDM-K64F

To run the Azure IoT Hub demo example, we need to update Azure IoT Hub MCUXpresso project with your Azure IoT Hub account details. This section explains what needs to be updated in Azure IoT Hub MCUXpresso project to add your Azure IoT Hub account details.

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.

3.9.1.1 Import Azure MCUXpresso example

The MCUXpresso projects are located in simw-top\projects folder. To import them into your MCUXpresso workspace, follow these steps:

1. Open a MCUXpresso workspace
2. Click **Import project(s) from file system** in the MCUXpresso IDE quick start panel as shown in **Figure 55**.

![Figure 55. Import projects from file system](image)

3. Click (1) **Browse** button in project directory (unpacked) section, (2) navigate to your `simw-top\projects` folder location as shown in **Figure 56** and (3) click **OK**:

![Figure 56. Browse project directory](image)

4. Click **Next** as shown in **Figure 57**:

![Figure 57](image)
5. Select (1) frdmk64f_mbedtls_sss_azure_demo, (2) unselect the Copy projects into workspace option, and (3) click Finish button as shown in Figure 58:
6. The projects you selected should be now visible into your MCUXpresso workspace as shown in Figure 59:

Figure 58. Select projects to be imported in the workspace
3.9.1.2 Change Azure IoT Hub project account settings

We need to change a few variables in the MCUXpresso Azure IoT Hub demo related with your Azure IoT Hub project account settings. In the MCUXpresso workspace:

1. Go to `frdmk64f_mbedtls_sss_azure_demo` project and open the `azure_iot_config.h` file as shown in Figure 60:
2. Replace the `#define AZURE_IOT_HUB_NAME` variable with your IoT Hub resource name created in Section 3.2 as shown Figure 61:

![Figure 61. Change AZURE_IOT_HUB_NAME variable](image1)

3. Replace the `#define AZURE_DEVICE_NAME` variable with your device resource ID created in Section 3.3 as shown Figure 62:

![Figure 62. Change AZURE_IOT_DEVICE_NAME variable](image2)

### 3.9.1.3 Start Azure IoT Hub demo

To debug the Azure IoT Hub demo, follow these steps:

1. Connect FRDM-K64F OpenSDA port, K64F port and Ethernet interface to your laptop as shown in Figure 63:
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:

![Configure TeraTerm](image)

3. Select the frdmk64f_mbedtls_sss_azure_demo project. Go to the MCUXpresso Quickstart Panel and click **Build** button as shown in Figure 65. Wait a few seconds until the project builds:
4. Go to the MCUXpresso Quickstart Panel and click **Debug** button as shown in **Figure 66**. If there is more than one probe attached, you have the select CMSIS-DAP from the list. Wait a few seconds until the project executes.

5. When it executes, it will automatically stop in a breakpoint. Click on **Resume** to allow the software to continue its execution.
6. Your device should now be connected to Azure. If it is connected successfully, the TeraTerm logs should look like Figure 68.

7. In your Azure IoT Hub instance dashboard, you can also check device activity statistics as shown in Figure 69.
8. You can also control the FRDM-K64F LEDs from the Azure dashboard. For instance, to turn on to green color the LED of FRDM-K64F, follow these steps:
   a. From the IoT Hub menu, go to IoT devices
   b. Click on your device name
   c. From the top menu, click on **Message to Device** button as shown in Figure 70.
d. Write the message `{"green": "on"}` inside the Message Body text box and click the Send Message button as shown in Figure 71:

![Figure 71. The FRDM-K64F LED turns green](image)

9. You can turn off the on-board LED of the FRDM-K64F writing the message `{"green": "off"}` inside the Message Body text box and click the Send Message button as shown in Figure 72:

![Figure 72. Turn off the FRDM-K64F on-board LED](image)

10. Finally, you can try to turn the LED of FRDM-K64F to blue color. For that, you just need to write `{"blue": "on"}` inside the Message Body text box and click the Send Message button as shown in Figure 73:

![Figure 73](image)
Figure 73. Turn the FRDM-K64F LED to blue color
4 Legal information

4.1 Definitions

Draft — The document is a draft version only. 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 herein and shall have no liability for the consequences of use of such information.

4.2 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 rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory. Notwithstanding any damages that NXP Semiconductors 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 planned, as well as for the planned application and use of customer’s third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products. 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 (US$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 — While NXP Semiconductors has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer’s applications and products, and NXP Semiconductors accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

4.3 Trademarks

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

Tab. 1. Abbreviations .............................................. 3  Tab. 3. FRDM-K64F details ........................................ 4
Tab. 2. OM-SE050ARD development kit details ........ 4
Figures

Fig. 1. Azure IoT Hub device registration flow .................................. 5
Fig. 2. Azure subscription creation ............................................. 7
Fig. 3. Azure free account sign up .............................................. 8
Fig. 4. Azure sign up verification by card ..................................... 9
Fig. 5. Azure sign up agreement acceptance .................................. 9
Fig. 6. Azure subscription completion ........................................ 10
Fig. 7. Create an Azure IoT Hub ............................................... 10
Fig. 8. Create an Azure IoT Hub - Basics form ............................. 11
Fig. 9. Create an Azure IoT Hub - Size and scale form .................... 12
Fig. 10. Create an Azure IoT Hub - Review and create form ............... 13
Fig. 11. Create an Azure IoT Hub completion ............................... 13
Fig. 12. Go to Azure IoT Hub resource ...................................... 14
Fig. 13. Create a device ......................................................... 14
Fig. 14. Create a device II ...................................................... 15
Fig. 15. Download Win32 OpenSSL binary .................................. 16
Fig. 16. OpenSSL license agreement ......................................... 16
Fig. 17. OpenSSL destination folder ........................................... 17
Fig. 18. OpenSSL start menu folder .......................................... 17
Fig. 19. OpenSSL DLLs directory ............................................. 18
Fig. 20. OpenSSL installation .................................................. 18
Fig. 21. OpenSSL installed ...................................................... 19
Fig. 22. Open your environment variables menu ............................ 19
Fig. 23. Set OPENSSL_CONF variable ..................................... 20
Fig. 24. Add openssl.exe location to the PATH variable .................. 21
Fig. 25. Check OpenSSL.exe environment variables .......................... 21
Fig. 26. PowerShell - Open as an Administrator ........................... 21
Fig. 27. PowerShell - Go to ca-certs.ps1 directory ......................... 22
Fig. 28. PowerShell - Set exution policy .................................... 22
Fig. 29. PowerShell - ca-certs.ps1 ............................................ 23
Fig. 30. PowerShell - test prerequisites ..................................... 23
Fig. 31. Manage user certificates ............................................. 24
Fig. 32. PowerShell - New-CaCertsCertChain ............................... 25
Fig. 33. PowerShell - Created certificates .................................... 25
Fig. 34. PowerShell - New-CaCertsDevice ................................. 26
Fig. 35. PowerShell - Created device certificate ........................... 26
Fig. 36. Go to Azure IoT Hub resource ...................................... 27
Fig. 37. Upload a root CA certificate ......................................... 27
Fig. 38. Root CA certificate unverified status ............................... 28
Fig. 39. Obtain root CA certificate verification status ..................... 28
Fig. 40. PowerShell - NewCaCertsVerificationCert ....................... 29
Fig. 41. PowerShell - verification certificate ................................ 29
Fig. 42. Root CA certificate verification ..................................... 30
Fig. 43. Root CA certificate verified ......................................... 30
Fig. 44. Unplug and plug OpenSDA port .................................... 31
Fig. 45. FRDM-K64F drive ..................................................... 32
Fig. 46. VCOM binary folder .................................................. 32
Fig. 47. Drag and drop VCOM binary ....................................... 33
Fig. 48. Check VCOM and serial ports ...................................... 34
Fig. 49. Credentials required .................................................. 34
Fig. 50. Start ssscli tool ....................................................... 35
Fig. 51. Close an already opened session ................................... 35
Fig. 52. Inject private key ...................................................... 35
Fig. 53. Inject certificate ....................................................... 36
Fig. 54. Disconnect ssscli ...................................................... 36
Fig. 55. Import projects from file system .................................... 37
Fig. 56. Browse project directory ........................................... 37
Fig. 57. Import projects into workspace .................................... 38
Fig. 58. Select projects to be imported in the workspace .................. 39
Fig. 59. Azure project in the workspace ................................... 40
Fig. 60. Open azure_iot_config.h file ...................................... 40
Fig. 61. Change AZURE_IOT_HUB_NAME variable ...................... 41
Fig. 62. Change AZURE_IOT_DEVICE_NAME variable ................. 41
Fig. 63. Connect FRDM-K64F board ......................................... 42
Fig. 64. Configure TeraTerm .................................................. 42
Fig. 65. Build Azure IoT Hub project ...................................... 43
Fig. 66. Debug Azure IoT Hub project ..................................... 43
Fig. 67. Resume Azure IoT Hub project .................................... 44
Fig. 68. TeraTerm logs - Device is connected to Azure IoT Hub ....... 44
Fig. 69. IoT Hub dashboard - Device is connected to Azure IoT Hub ... 45
Fig. 70. Click on Message to Device button ............................... 45
Fig. 71. The FRDM-K64F LED turns green ................................ 46
Fig. 72. Turn off the FRDM-K64F on-board LED ......................... 46
Fig. 73. Turn the FRDM-K64F LED to blue color ......................... 47
Contents

1 Read this first ......................................................4
1.1 Prerequisites ......................................................4
1.2 Required hardware ............................................ 4
1.3 Required software ............................................. 4
2 Azure IoT Hub device registration flow .............5
3 Azure IoT Hub demo example ...........................7
3.1 Create Azure subscription ..................................7
3.2 Create an Azure IoT Hub ...................................10
3.3 Create a device ...............................................14
3.4 Create your own X.509 certificates ............... 15
3.4.1 Create your own X.509 certificates using PowerShell .......................................................15
3.4.1.1 Install OpenSSL ...............................................15
3.4.1.2 Set OpenSSL environment variables ...............19
3.4.1.3 Initial setup ...................................................... 21
3.4.1.4 Create the certificate chain of trust ..........24
3.4.1.5 Create device certificate ..............................25
3.5 Upload root CA certificate to your Azure IoT Hub ............................................................. 26
3.6 Create verification certificate ......................... 28
3.6.1 Create verification certificate using PowerShell ............................................................. 29
3.7 Verify root CA certificate ..................................30
3.8 Key provisioning ............................................ 30
3.8.1 Key provisioning using FRDM-K64F ..........31
3.8.1.1 Flash FRDM-K64F with VCOM software .........31
3.8.1.2 Key injection .................................................... 34
3.9 Running Azure IoT Hub demo ....................... 36
3.9.1 Running Azure IoT Hub demo using FRDM-K64F ............................................................. 36
3.9.1.1 Import Azure MCUXpresso example ..........36
3.9.1.2 Change Azure IoT Hub project account settings ............................................................. 40
3.9.1.3 Start Azure IoT Hub demo .......................... 41
4 Legal information ..............................................48