AN12402

EdgeLock^{TM} SE05x for secure connection to Azure IoT Hub

Rev. 2.1 — 7 December 2020 Application note
535020

Document information

Information	Content
Keywords	EdgeLock SE05x, Azure IoT Hub, Secure cloud onboarding
Abstract	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.

Revision history

Revision history

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

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.

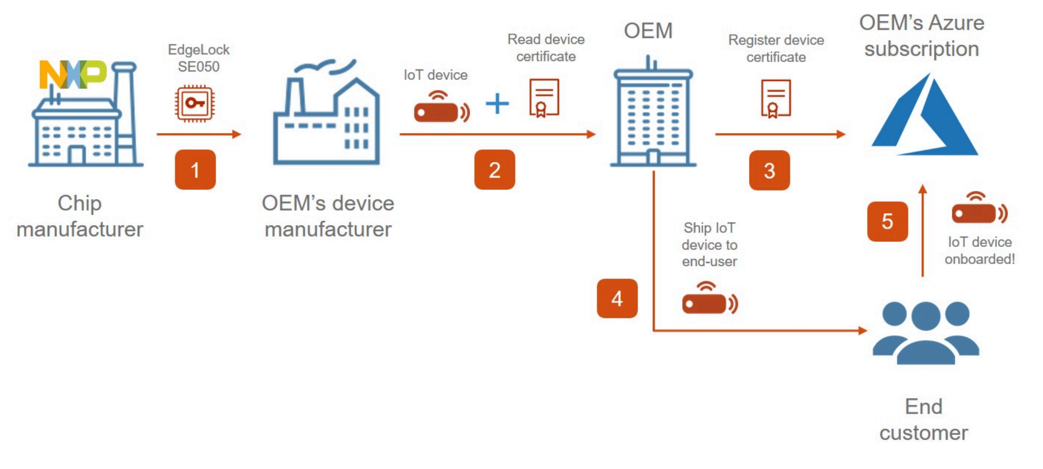


Figure 1. Azure IoT Hub device registration flow

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.

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.

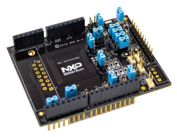
Note: *The Azure IoT Hub device onboarding procedure described in this section and the Azure IoT Hub demo example are provided only for evaluation purposes. Therefore, the subsequent procedure must be adapted and adjusted accordingly for a commercial deployment.*

3.1 Hardware required

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

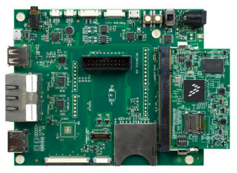
1. OM-SE050ARD development kit:

Table 1. OM-SE050ARD development kit details

Part number	12NC	Content	Picture
OM-SE050ARD	935383282598	EdgeLock SE050 development board	

i.MX6Ultralite board

Table 2. i.MX6Ultralite

Part number	12NC	Content	Picture
MCIMX6UL-EVKB	935328353598	i.MX6UltraLite evaluation kit	

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

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

Key name and type	Certificate	Usage policy	Erasable by customer	Identifier
Cloud connection key 0, ECC256, Die Individual	Cloud Connectivity Certificate 0, ECC signed	Default	Yes	Key: 0xF0000100 Cert: 0xF0000101

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

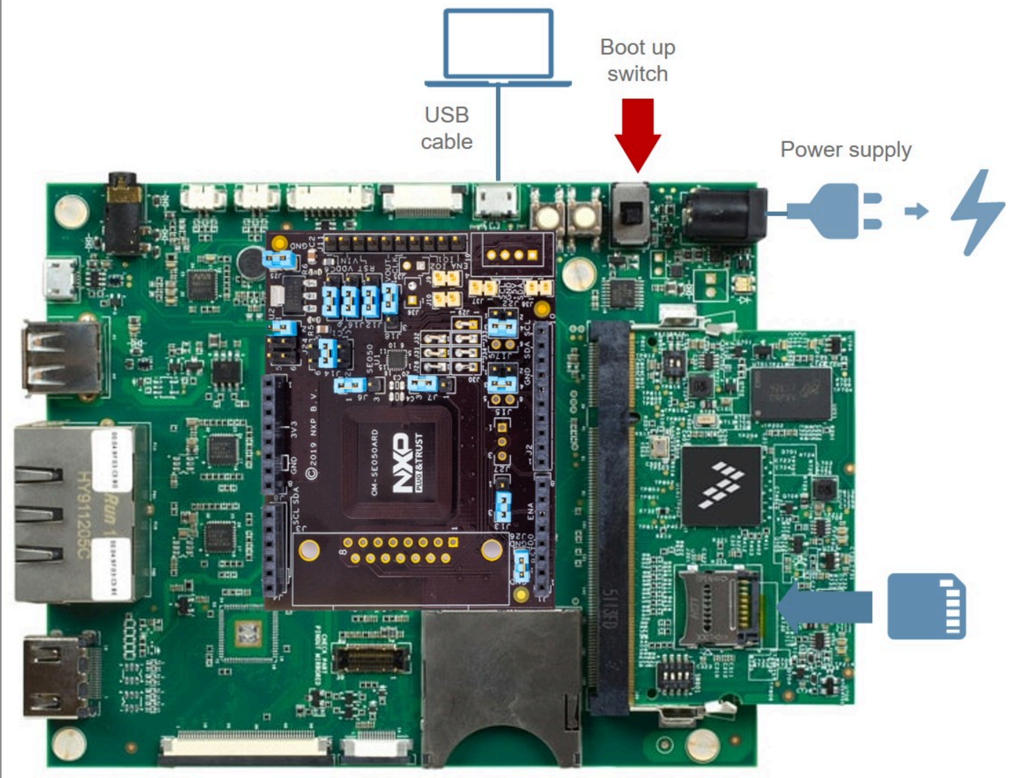
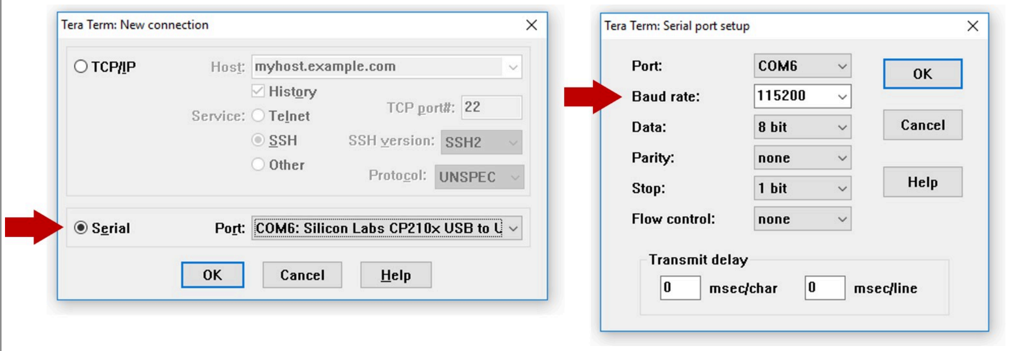


Figure 2. Boot up i.MX6UltraLite board.

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:



Tera Term: New connection

TCP/IP Host: myhost.example.com
History
Service: Telnet TCP port#: 22
SSH SSH version: SSH2
Other Protocol: UNSPEC

Serial Port: COM6: Silicon Labs CP210x USB to U

OK Cancel Help

Tera Term: Serial port setup

Port:	COM6
Baud rate:	115200
Data:	8 bit
Parity:	none
Stop:	1 bit
Flow control:	none

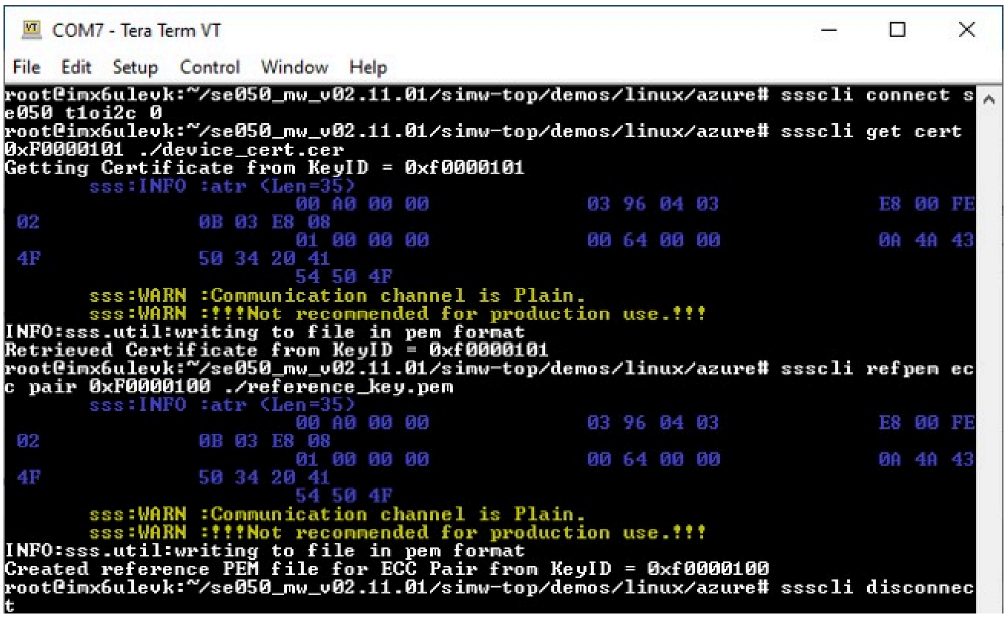
Transmit delay
0 msec/char 0 msec/line

OK Cancel Help

Figure 3. Configure TeraTerm

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`.

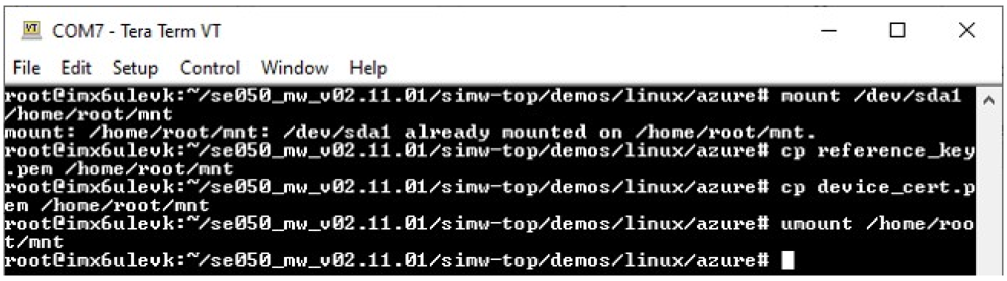
4. Follow these steps to open the `ssscli` tool and retrieve the device certificate and the private key reference. Figure 4 shows the execution of these steps.
   a. Send > `ssscli connect se050 t1oi2c 0`.
   b. Send > `ssscli get cert 0xF0000101 ./device_cert.cer`.
   c. Send > `ssscli refpem ecc pair 0xF0000100 ./reference_key.pem`.
   d. Send > `ssscli disconnect`.



```
root@imx6ulevk:~/se050_mw_v02.11.01/simw-top/demos/linux/azure# ssscli connect s
e050 t1oi2c 0
root@imx6ulevk:~/se050_mw_v02.11.01/simw-top/demos/linux/azure# ssscli get cert
0xF0000101 ./device_cert.cer
Getting Certificate from KeyID = 0xf0000101
       sss:INFO :atr (Len=35)
                 00 A0 00 00          03 96 04 03          E8 00 FE
 02              0B 03 E8 08
                 01 00 00 00          00 64 00 00          0A 4A 43
 4F              50 34 20 41
                 54 50 4F
       sss:WARN :Communication channel is Plain.
       sss:WARN :!!!Not recommended for production use.!!!
INFO:sss.util:writing to file in pem format
Retrieved Certificate from KeyID = 0xf0000101
root@imx6ulevk:~/se050_mw_v02.11.01/simw-top/demos/linux/azure# ssscli refpem ec
c pair 0xF0000100 ./reference_key.pem
       sss:INFO :atr (Len=35)
                 00 A0 00 00          03 96 04 03          E8 00 FE
 02              0B 03 E8 08
                 01 00 00 00          00 64 00 00          0A 4A 43
 4F              50 34 20 41
                 54 50 4F
       sss:WARN :Communication channel is Plain.
       sss:WARN :!!!Not recommended for production use.!!!
INFO:sss.util:writing to file in pem format
Created reference PEM file for ECC Pair from KeyID = 0xf0000100
root@imx6ulevk:~/se050_mw_v02.11.01/simw-top/demos/linux/azure# ssscli disconnec
t
```

Figure 4. Execution of ssscli commands to retrieve credentials.

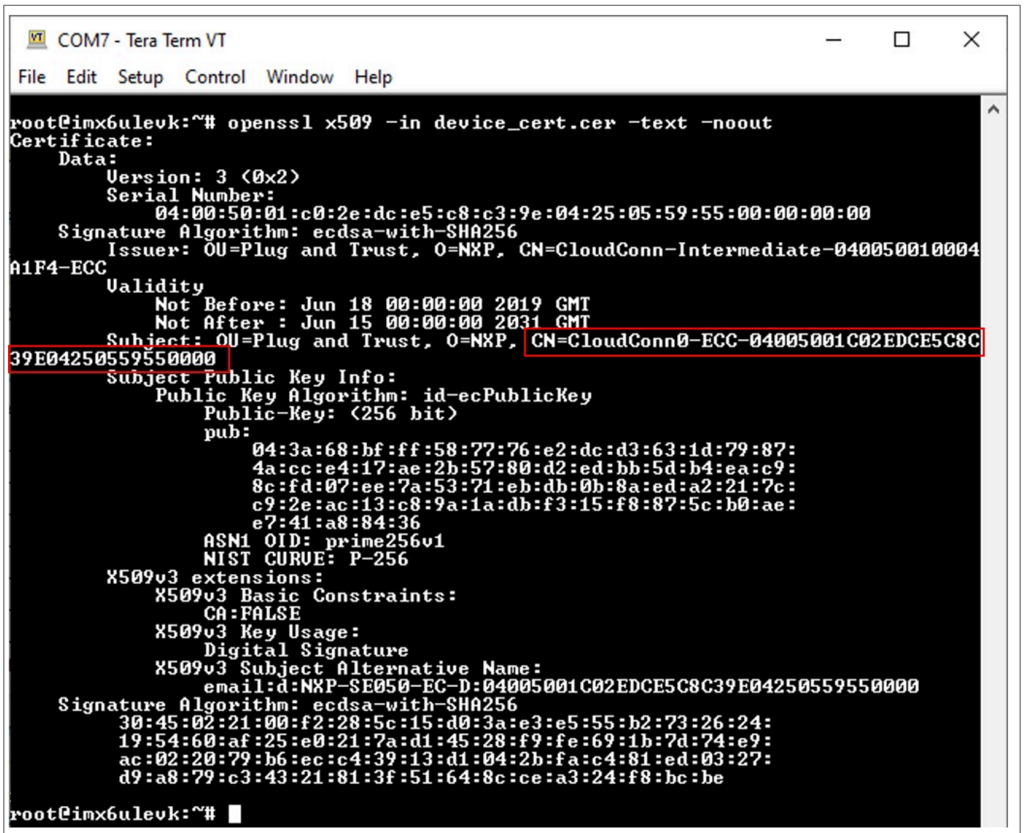
5. Once the credentials have been extracted from the EdgeLock SE05x, we need to upload them to our Azure Device Provisioning Service instance. To do so, copy them in a USB device following the steps shown in Figure 5:
   a. Connect a USB device in the available port in the i.MX6UltraLite board.
   b. 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`.



```
root@imx6ulevk:~/se050_mw_v02.11.01/simw-top/demos/linux/azure# mount /dev/sda1
/home/root/mnt
mount: /home/root/mnt: /dev/sda1 already mounted on /home/root/mnt.
root@imx6ulevk:~/se050_mw_v02.11.01/simw-top/demos/linux/azure# cp reference_key
.pem /home/root/mnt
root@imx6ulevk:~/se050_mw_v02.11.01/simw-top/demos/linux/azure# cp device_cert.p
em /home/root/mnt
root@imx6ulevk:~/se050_mw_v02.11.01/simw-top/demos/linux/azure# umount /home/roo
t/mnt
root@imx6ulevk:~/se050_mw_v02.11.01/simw-top/demos/linux/azure# 
```

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



```
root@imx6ulevk:~# openssl x509 -in device_cert.cer -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            04:00:50:01:c0:2e:dc:e5:c8:c3:9e:04:25:05:59:55:00:00:00:00
    Signature Algorithm: ecdsa-with-SHA256
        Issuer: OU=Plug and Trust, O=NXP, CN=CloudConn-Intermediate-040050010004
A1F4-ECC
        Validity
            Not Before: Jun 18 00:00:00 2019 GMT
            Not After : Jun 15 00:00:00 2031 GMT
        Subject: OU=Plug and Trust, O=NXP, CN=CloudConn0-ECC-04005001C02EDCE5C8C
39E04250559550000
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    04:3a:68:bf:ff:58:77:76:e2:dc:d3:63:1d:79:87:
                    4a:cc:e4:17:ae:2b:57:80:d2:ed:bb:5d:b4:ea:c9:
                    8c:fd:07:ee:7a:53:71:eb:db:0b:8a:ed:a2:21:7c:
                    c9:2e:ac:13:c8:9a:1a:db:f3:15:f8:87:5c:b0:ae:
                    e7:41:a8:84:36
                ASN1 OID: prime256v1
                NIST CURVE: P-256
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Key Usage:
                Digital Signature
            X509v3 Subject Alternative Name:
                email:d:NXP-SE050-EC-D:04005001C02EDCE5C8C39E04250559550000
    Signature Algorithm: ecdsa-with-SHA256
         30:45:02:21:00:f2:28:5c:15:d0:3a:e3:e5:55:b2:73:26:24:
         19:54:60:af:25:e0:21:7a:d1:45:28:f9:fe:69:1b:7d:74:e9:
         ac:02:20:79:b6:ec:c4:39:13:d1:04:2b:fa:c4:81:ed:03:27:
         d9:a8:79:c3:43:21:81:3f:51:64:8c:ce:a3:24:f8:bc:be

root@imx6ulevk:~#
```

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:

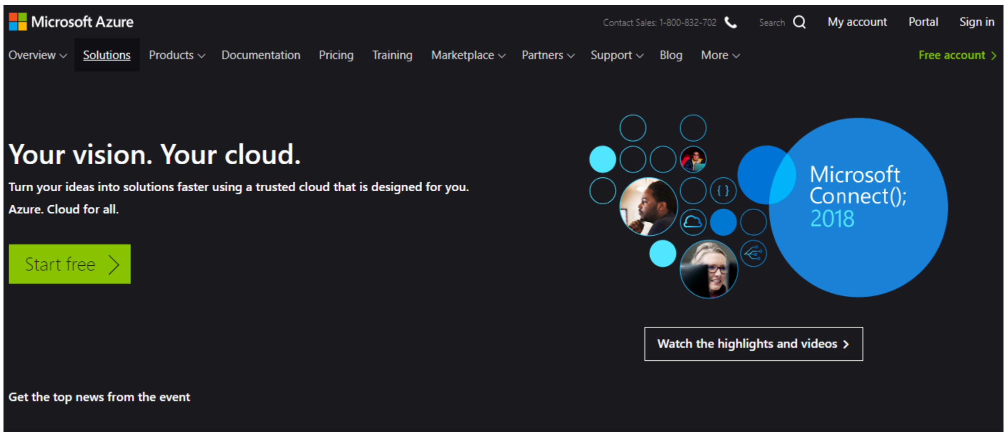
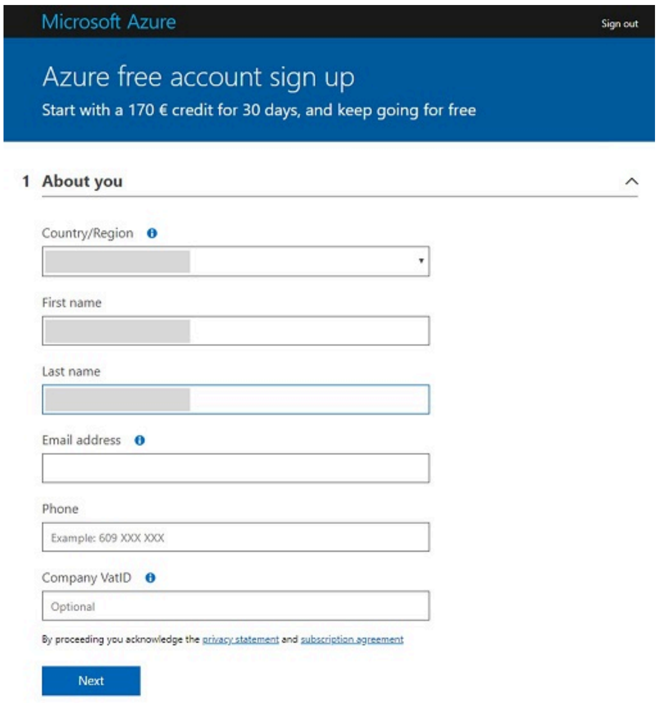

Microsoft Azure
Contact Sales: 1-800-832-702
Search
My account
Portal
Sign in
Overview
Solutions
Products
Documentation
Pricing
Training
Marketplace
Partners
Support
Blog
More
Free account
Your vision. Your cloud.
Turn your ideas into solutions faster using a trusted cloud that is designed for you.
Azure. Cloud for all.
Start free
Microsoft Connect(); 2018
Watch the highlights and videos
Get the top news from the event

Figure 7. Azure subscription creation

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 8:



Azure free account sign up

Start with a 170 € credit for 30 days, and keep going for free

1 About you

Country/Region

First name

Last name

Email address

Phone

Example: 609 XXX XXX

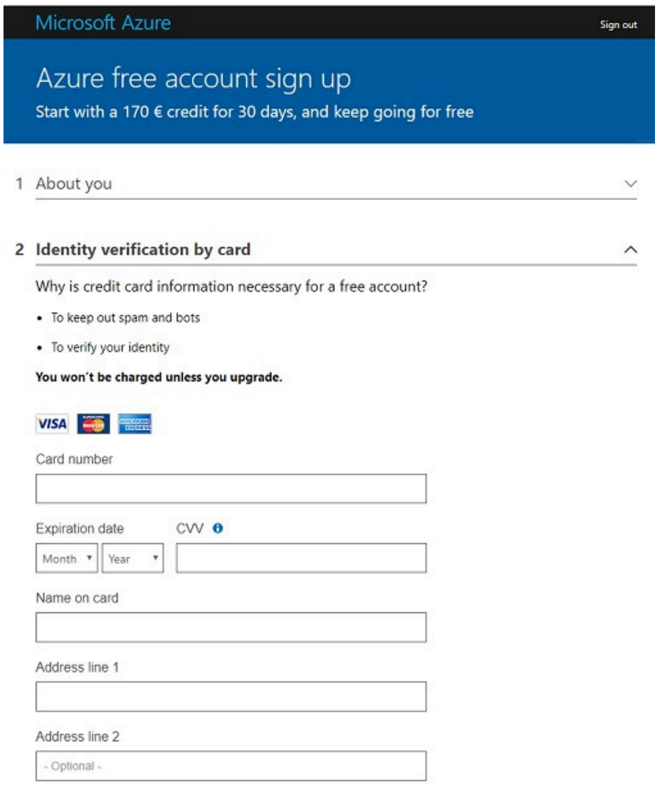
Company VatID

Optional

By proceeding you acknowledge the privacy statement and subscription agreement

Figure 8. Azure free account sign up

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.



Microsoft Azure
Sign out

Azure free account sign up

Start with a 170 € credit for 30 days, and keep going for free

1 About you

2 Identity verification by card

Why is credit card information necessary for a free account?

- To keep out spam and bots
- To verify your identity

You won't be charged unless you upgrade.

Card number

Expiration date CVV

Month Year

Name on card

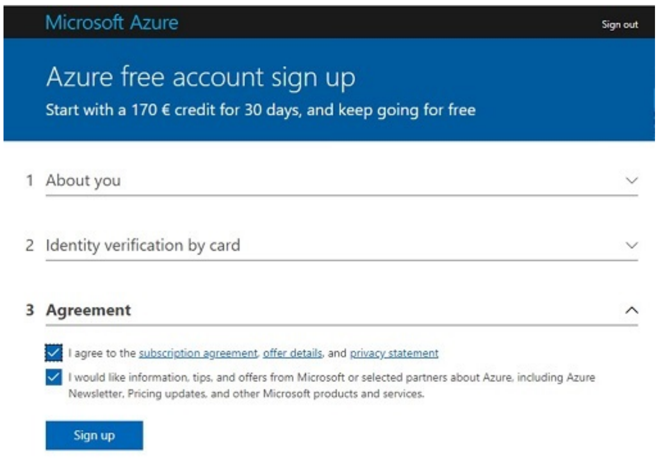
Address line 1

Address line 2

- Optional -

Figure 9. Azure sign up verification by card

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



Microsoft Azure
Sign out

Azure free account sign up

Start with a 170 € credit for 30 days, and keep going for free

1 About you

2 Identity verification by card

3 Agreement

✓ I agree to the subscription agreement, offer details, and privacy statement

✓ I would like information, tips, and offers from Microsoft or selected partners about Azure, including Azure Newsletter, Pricing updates, and other Microsoft products and services.

Sign up

Figure 10. Azure sign up agreement acceptance

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:

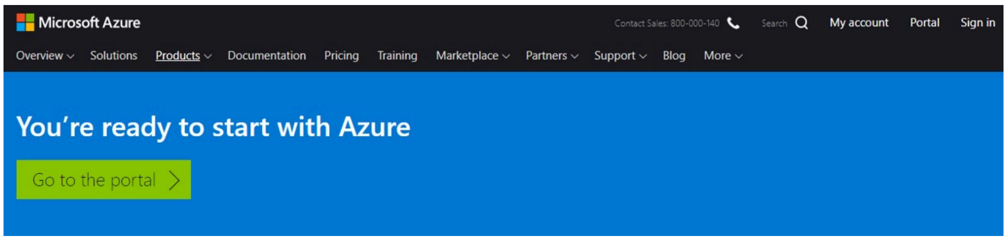


Figure 11. Azure subscription completition

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. In the Azure dashboard do as shown in Figure 12:
   a. Choose ***Create resource***.
   b. Choose ***Internet of Things***
   c. Choose ***IoT Hub***

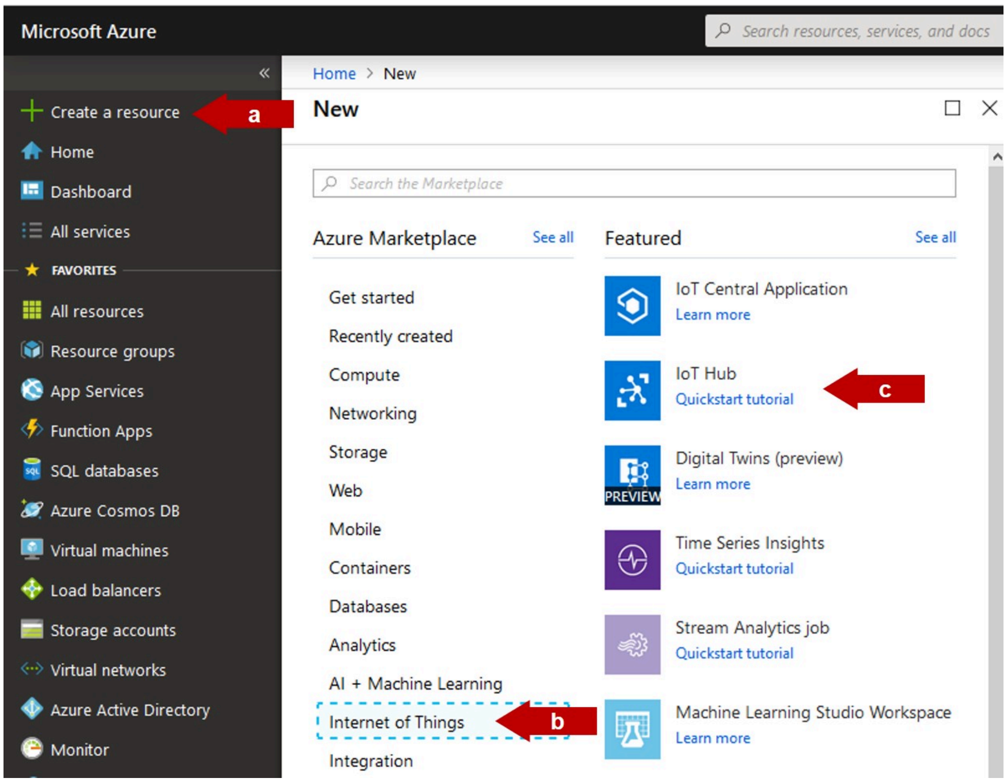
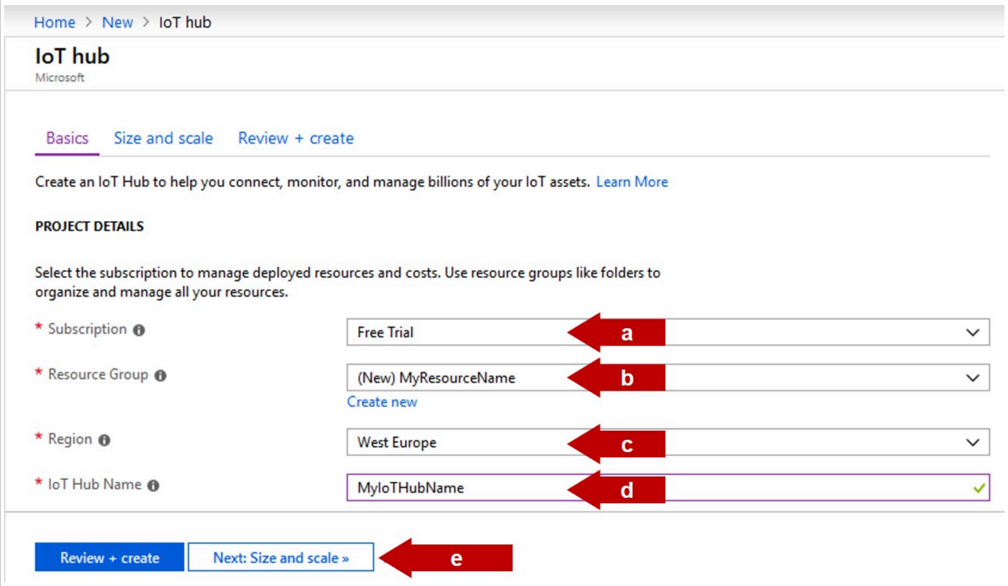


Figure 12. Create an Azure IoT Hub

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 acount 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:***


Home > New > IoT hub

IoT hub

Microsoft

Basics | Size and scale | Review + create

Create an IoT Hub to help you connect, monitor, and manage billions of your IoT assets. Learn More

PROJECT DETAILS

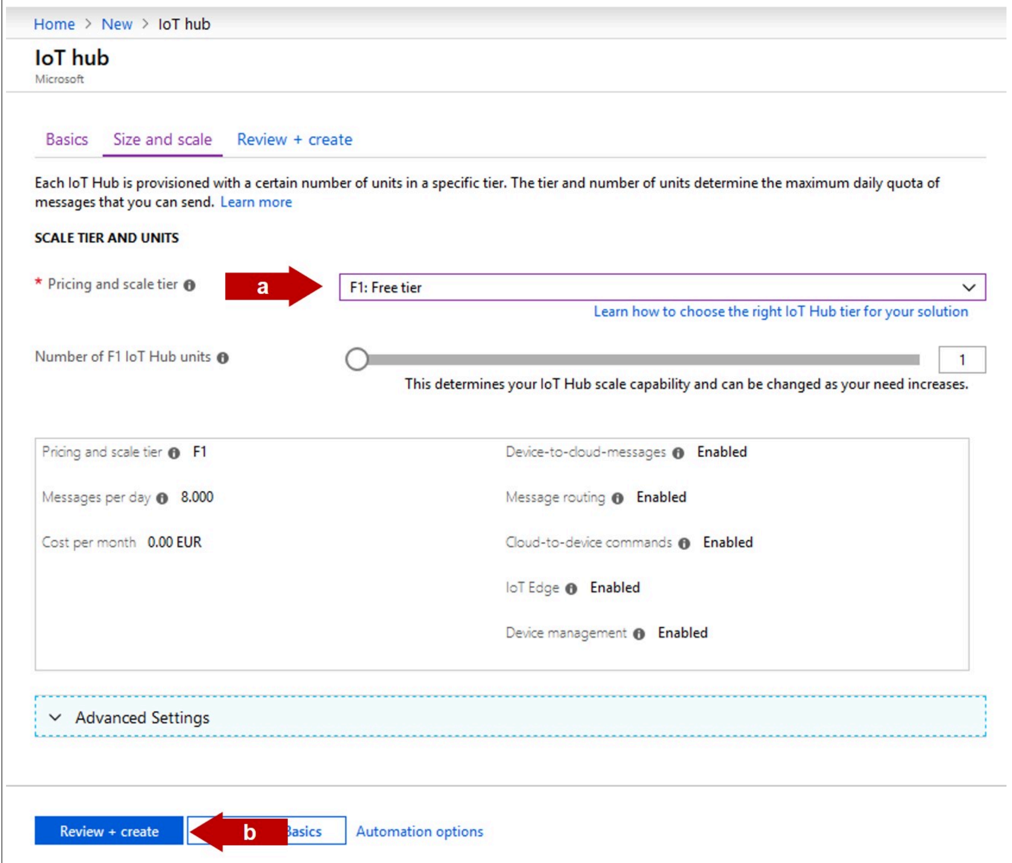
Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

* Subscription	Free Trial
* Resource Group	(New) MyResourceName
	Create new
* Region	West Europe
* IoT Hub Name	MyIoTHubName

Figure 13. Create an Azure IoT Hub - Basics form

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.



Each IoT Hub is provisioned with a certain number of units in a specific tier. The tier and number of units determine the maximum daily quota of messages that you can send. Learn more

SCALE TIER AND UNITS

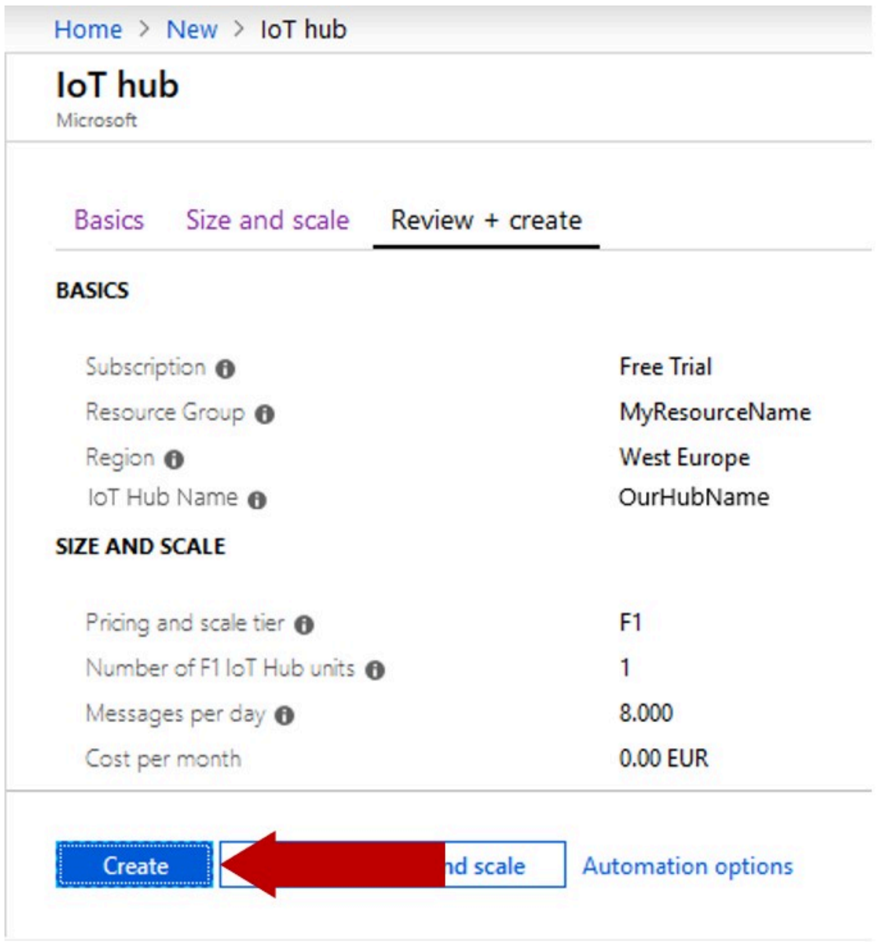
* Pricing and scale tier	F1: Free tier
Number of F1 IoT Hub units	1

This determines your IoT Hub scale capability and can be changed as your need increases.

Pricing and scale tier	F1
Messages per day	8.000
Cost per month	0.00 EUR
Device-to-cloud-messages	Enabled
Message routing	Enabled
Cloud-to-device commands	Enabled
IoT Edge	Enabled
Device management	Enabled

Figure 14. Create an Azure IoT Hub - Size and scale form

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:



BASICS

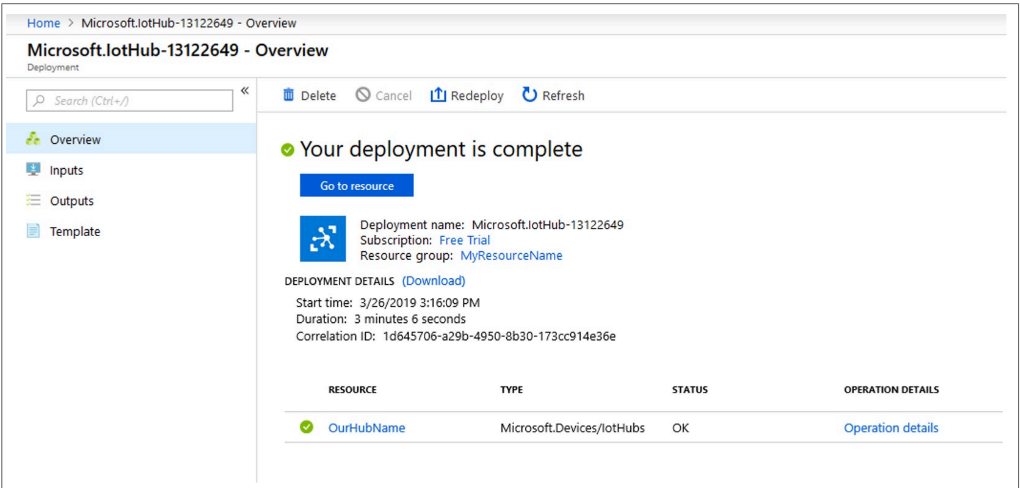
Subscription	Free Trial
Resource Group	MyResourceName
Region	West Europe
IoT Hub Name	OurHubName

SIZE AND SCALE

Pricing and scale tier	F1
Number of F1 IoT Hub units	1
Messages per day	8.000
Cost per month	0.00 EUR

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:



Your deployment is complete

Deployment name: Microsoft.IotHub-13122649
Subscription: Free Trial
Resource group: MyResourceName

DEPLOYMENT DETAILS (Download)

Start time: 3/26/2019 3:16:09 PM
Duration: 3 minutes 6 seconds
Correlation ID: 1d645706-a29b-4950-8b30-173cc914e36e

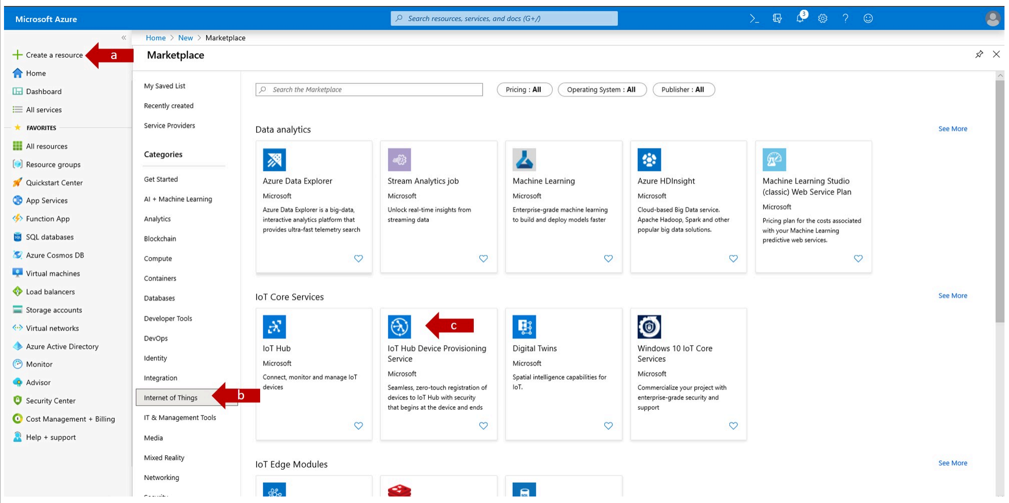
RESOURCE	TYPE	STATUS	OPERATION DETAILS
✓ OurHubName	Microsoft.Devices/IotHubs	OK	Operation details

Figure 16. Create an Azure IoT Hub completition

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***.



Azure Data Explorer
Microsoft
Azure Data Explorer is a big-data, interactive analytics platform that provides ultra-fast telemetry search

Stream Analytics job
Microsoft
Unlock real-time insights from streaming data

Machine Learning
Microsoft
Enterprise-grade machine learning to build and deploy models faster

Azure HDInsight
Microsoft
Cloud-based Big Data service. Apache Hadoop, Spark and other popular big data solutions.

Machine Learning Studio (classic) Web Service Plan
Microsoft
Pricing plan for the costs associated with your Machine Learning predictive web services.

IoT Hub
Microsoft
Connect, monitor and manage IoT devices

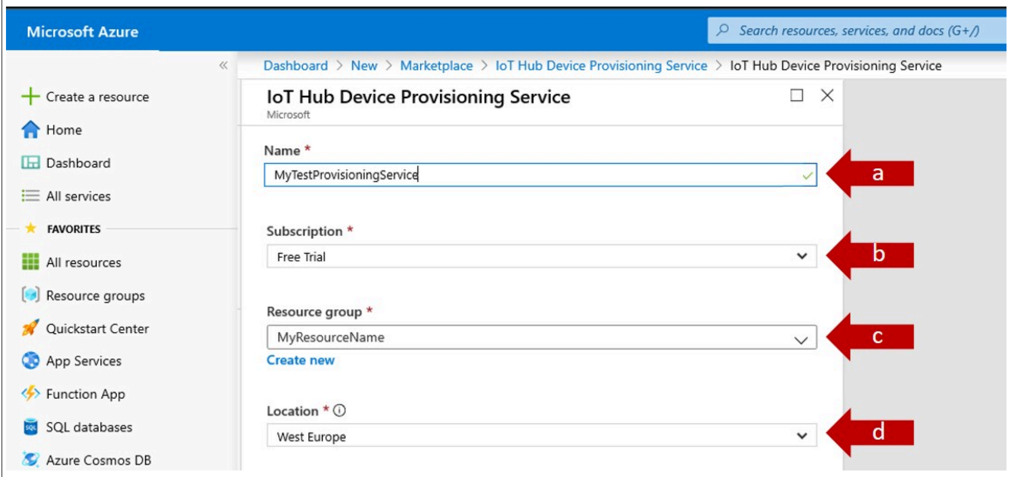
IoT Hub Device Provisioning Service
Microsoft
Seamless, zero-touch registration of devices to IoT Hub with security that begins at the device and ends

Digital Twins
Microsoft
Spatial intelligence capabilities for IoT.

Windows 10 IoT Core Services
Microsoft
Commercialize your project with enterprise-grade security and support

Figure 17. Create an IoT Hub Device Provisioning Service

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.



IoT Hub Device Provisioning Service
Microsoft

Name *
MyTestProvisioningService

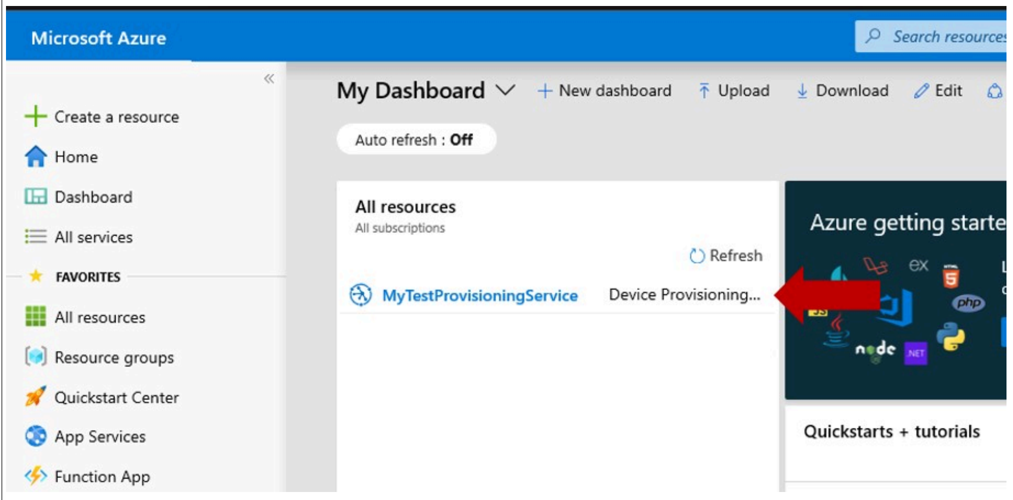
Subscription *
Free Trial

Resource group *
MyResourceName
Create new

Location *
West Europe

Figure 18. Provide the basic information of the IoT Hub Device Provisioning Service

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

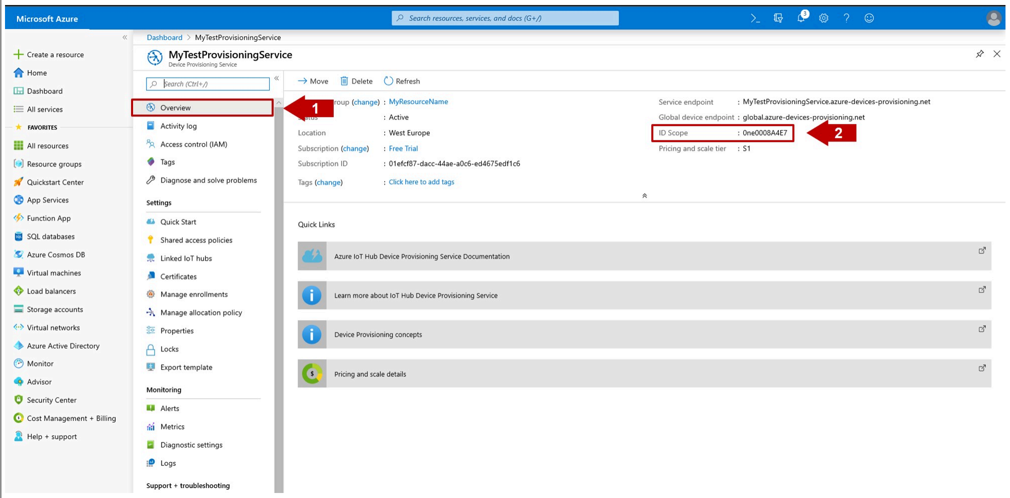


All resources
All subscriptions

MyTestProvisioningService Device Provisioning...

Figure 19. IoT Hub Device Provisioning Service instance in Azure Dashboard

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.


Microsoft Azure
Search resources, services, and docs (G+/)
Dashboard > MyTestProvisioningService
MyTestProvisioningService
Device Provisioning Service
Overview
1
2
Move
Delete
Refresh
Quick Links

Resource group (change) : MyResourceName
Status : Active
Location : West Europe
Subscription (change) : Free Trial
Subscription ID : 01efcf87-dacc-44ae-a0c6-ed4675edf1c6
Tags (change) : Click here to add tags
Service endpoint : MyTestProvisioningService.azure-devices-provisioning.net
Global device endpoint : global.azure-devices-provisioning.net
ID Scope : 0ne0008A4E7
Pricing and scale tier : S1

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.

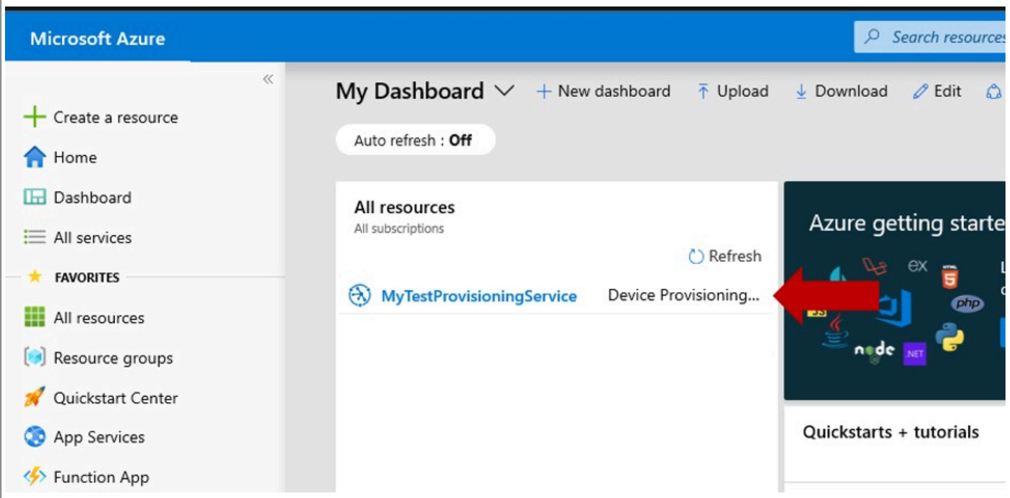
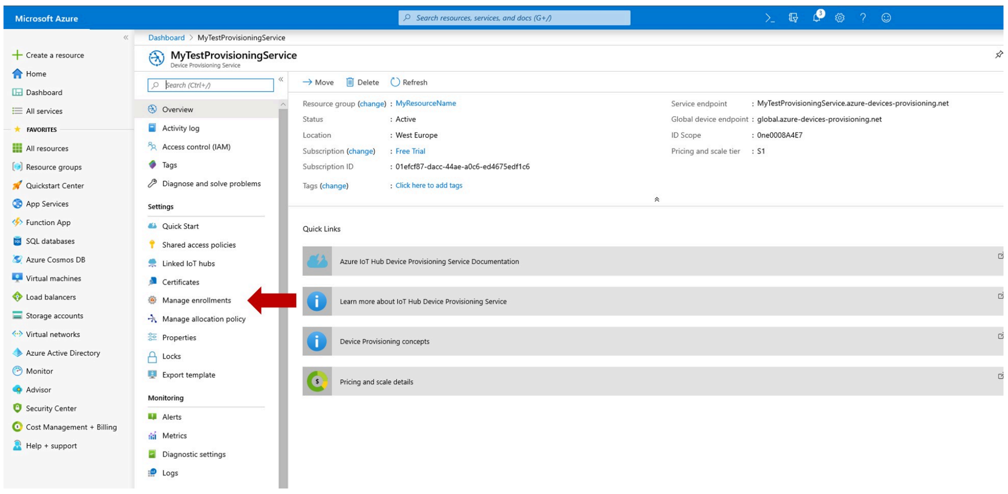


Figure 21. IoT Hub Device Provisioning Service instance in Azure Dashboard.

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

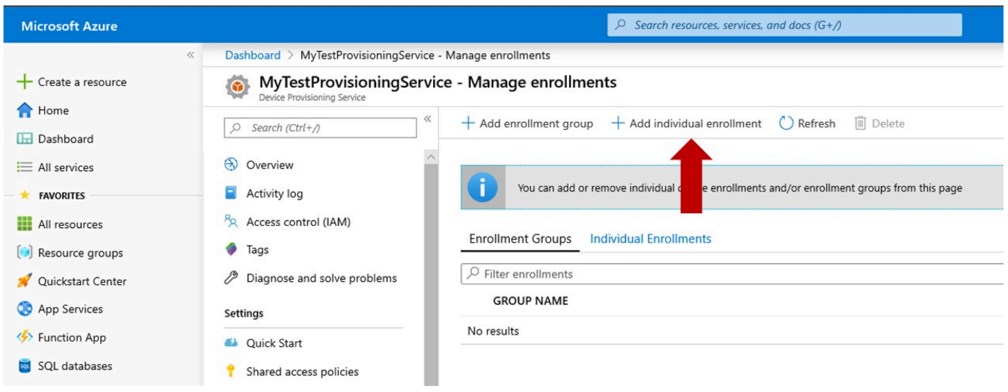


Resource group (change) : MyResourceName
Status : Active
Location : West Europe
Subscription (change) : Free Trial
Subscription ID : 01efcf87-dacc-44ae-a0c6-ed4675edf1c6
Tags (change) : Click here to add tags

Service endpoint : MyTestProvisioningService.azure-devices-provisioning.net
Global device endpoint : global.azure-devices-provisioning.net
ID Scope : 0ne0008A4E7
Pricing and scale tier : S1

Figure 22. Overview of the IoT Hub Device Provisioning Service instance.

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

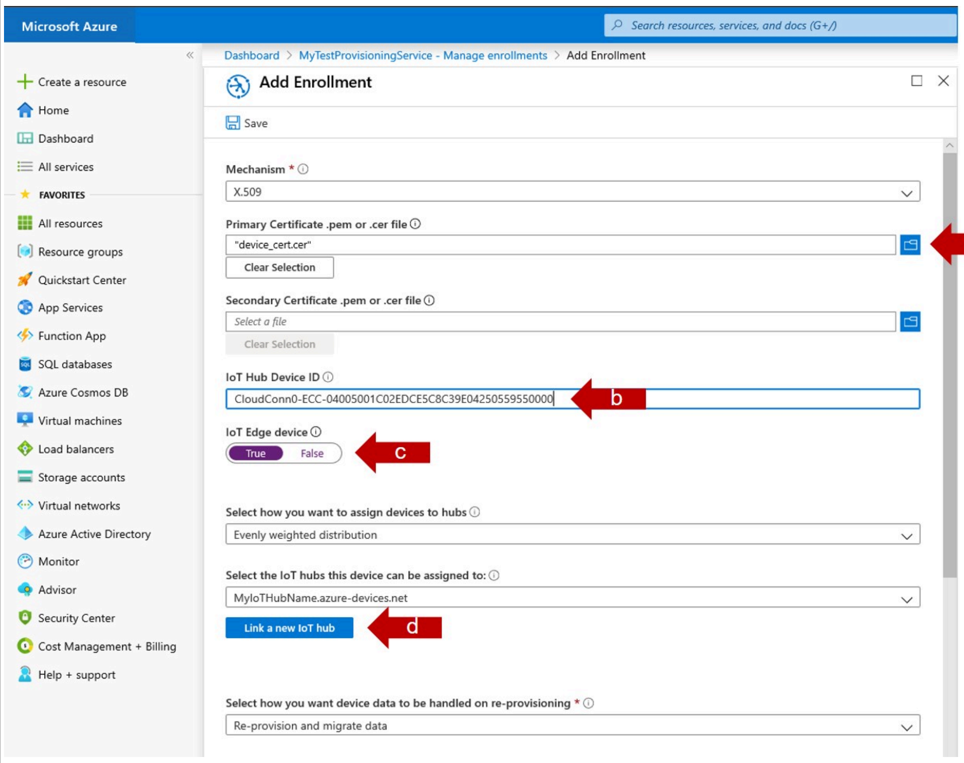


You can add or remove individual d e enrollments and/or enrollment groups from this page

No results

Figure 23. Manage enrollments.

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.



Add Enrollment

Mechanism *: X.509

Primary Certificate .pem or .cer file: "device_cert.cer"

Secondary Certificate .pem or .cer file: Select a file

IoT Hub Device ID: CloudConn0-ECC-04005001C02EDCE5C8C39E04250559550000

IoT Edge device: True

Select how you want to assign devices to hubs: Evenly weighted distribution

Select the IoT hubs this device can be assigned to: MyIoTHubName.azure-devices.net

Select how you want device data to be handled on re-provisioning *: Re-provision and migrate data

Figure 24. Add individual enrollment detail.

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:

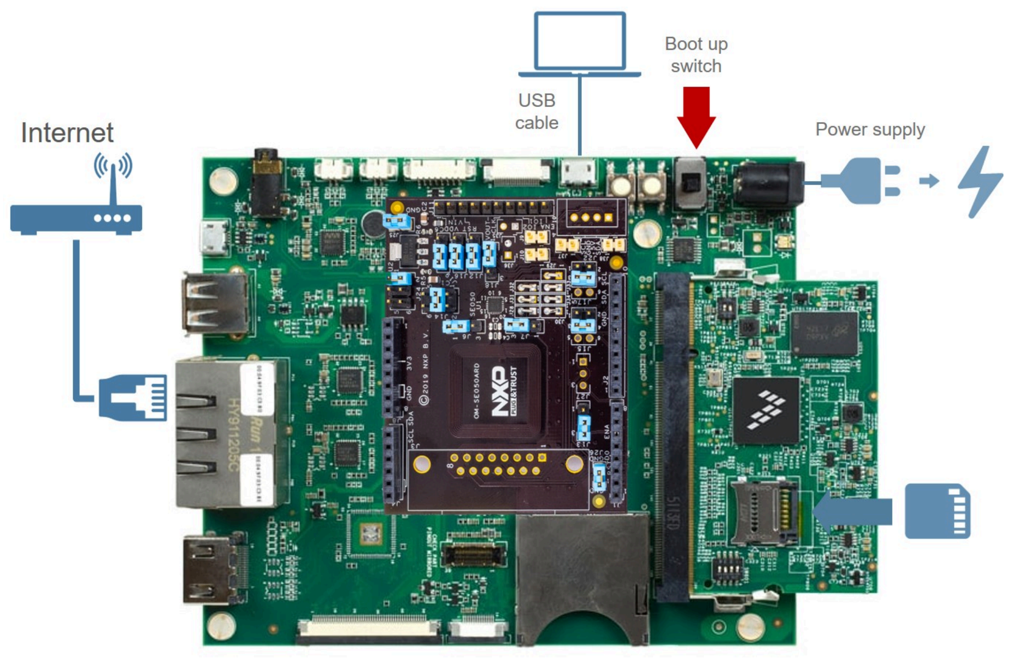
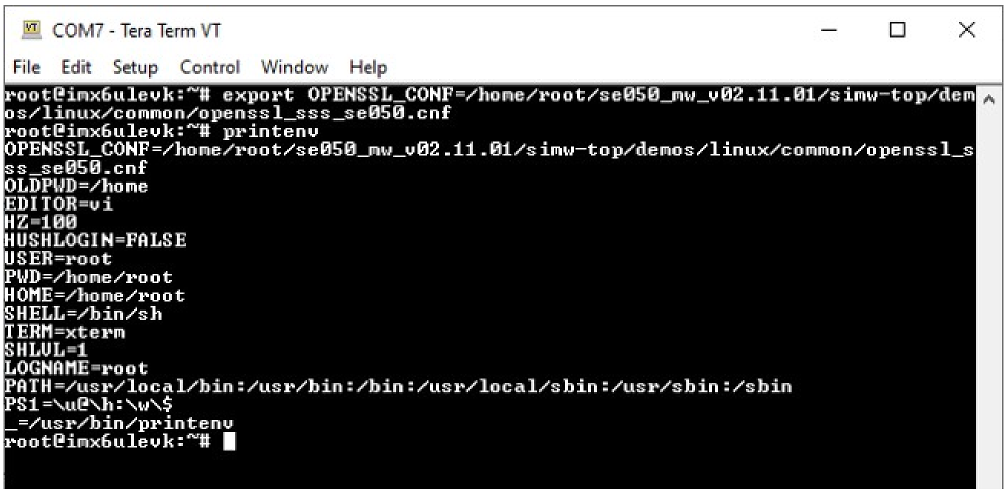


Figure 25. Connect board to Internet

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`.



```
root@imx6ulevk:~# export OPENSSL_CONF=/home/root/se050_mw_v02.11.01/simw-top/dem
os/linux/common/openssl_sss_se050.cnf
root@imx6ulevk:~# printenv
OPENSSL_CONF=/home/root/se050_mw_v02.11.01/simw-top/demos/linux/common/openssl_s
ss_se050.cnf
OLDPWD=/home
EDITOR=vi
HZ=100
HUSHLOGIN=FALSE
USER=root
PWD=/home/root
HOME=/home/root
SHELL=/bin/sh
TERM=xterm
SHLVL=1
LOGNAME=root
PATH=/usr/local/bin:/usr/bin:/bin:/usr/local/sbin:/usr/sbin:/sbin
PS1=\u@\h:\w\$
_=/usr/bin/printenv
root@imx6ulevk:~# 
```

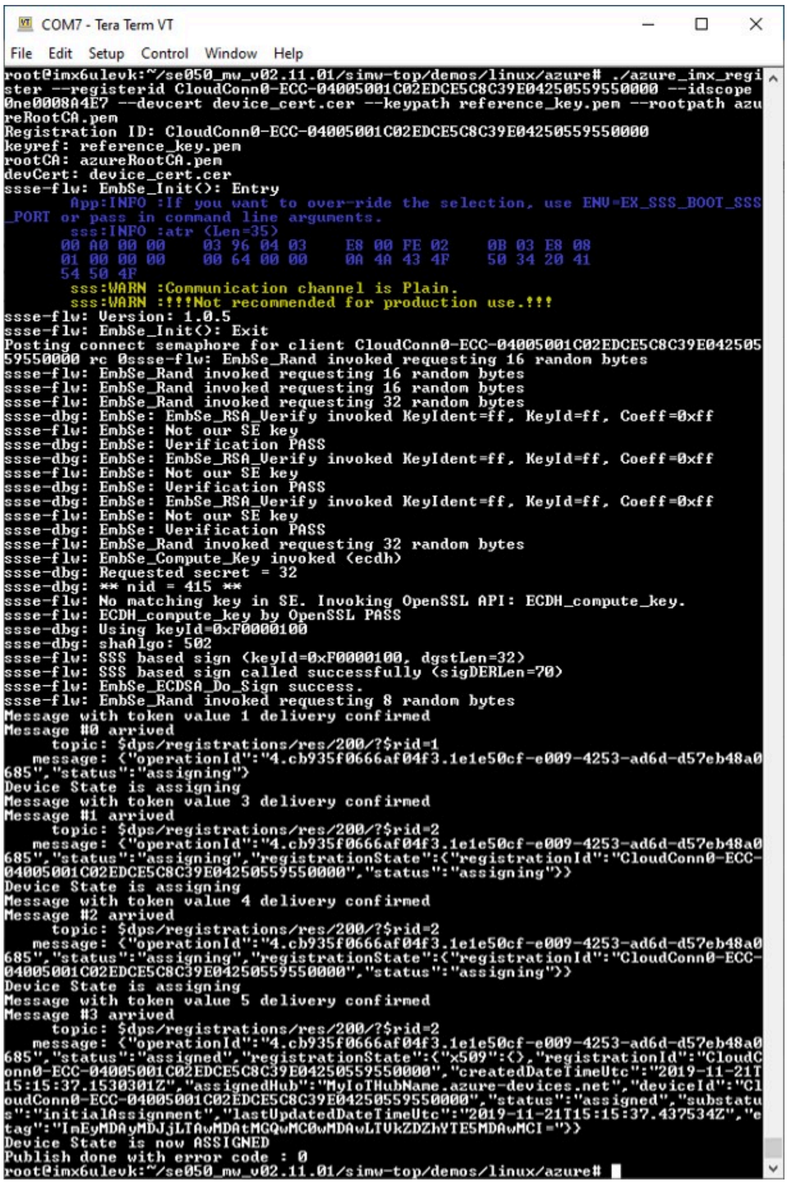
Figure 26. Export the OPENSSL_CONF variable.

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-04005001C02EDCE5C8C39E04250559550000 --idscope 0ne0008A4E7`

`--devcert device_cert.cer --keypath reference_key.pem --rootpath azureRootCA.pem`.

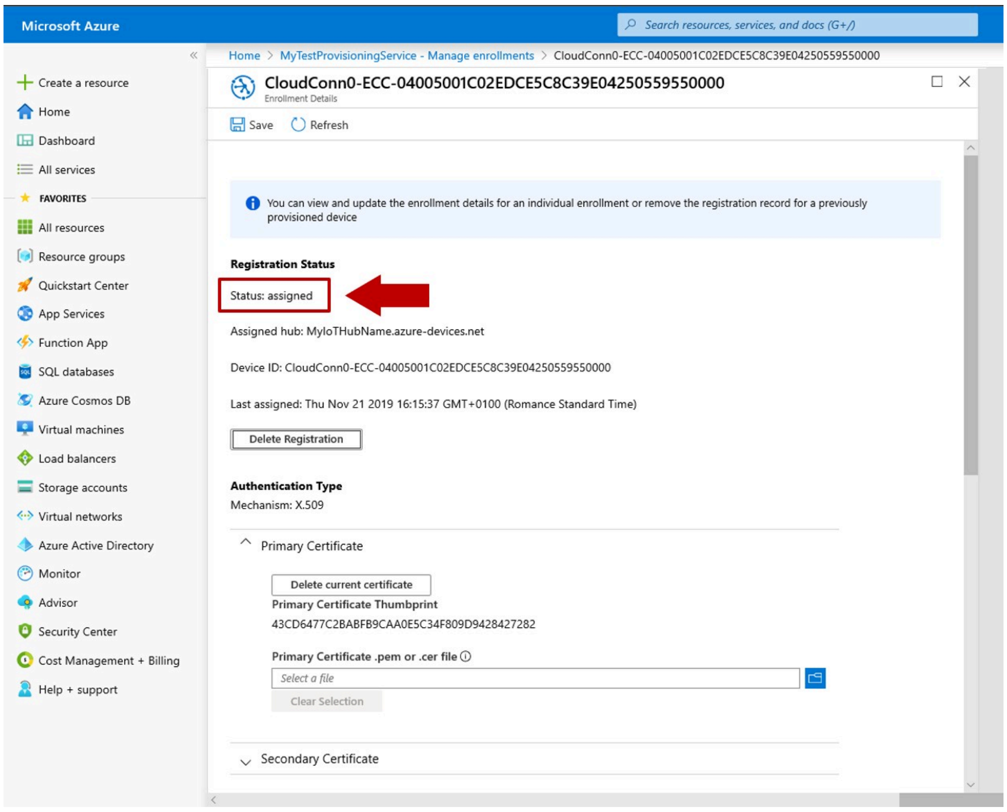
Figure 27 shows the output of running the `azure_imx_register` script.



```
root@imx6ulevk:~/se050_mw_v02.11.01/simw-top/demos/linux/azure# ./azure_imx_regi
ster --registerid CloudConn0-ECC-04005001C02EDCE5C8C39E04250559550000 --idscope
0ne0008A4E7 --devcert device_cert.cer --keypath reference_key.pem --rootpath azu
reRootCA.pem
Registration ID: CloudConn0-ECC-04005001C02EDCE5C8C39E04250559550000
keyref: reference_key.pem
rootCA: azureRootCA.pem
devCert: device_cert.cer
ssse-flw: EmbSe_Init(): Entry
       App:INFO :If you want to over-ride the selection, use ENV=EX_SSS_BOOT_SSS
_PORT or pass in command line arguments.
       sss:INFO :atr (Len=35)
      00 A0 00 00    03 96 04 03    E8 00 FE 02    0B 03 E8 08
      01 00 00 00    00 64 00 00    0A 4A 43 4F    50 34 20 41
      54 50 4F
       sss:WARN :Communication channel is Plain.
       sss:WARN :!!!Not recommended for production use.!!!
ssse-flw: Version: 1.0.5
ssse-flw: EmbSe_Init(): Exit
Posting connect semaphore for client CloudConn0-ECC-04005001C02EDCE5C8C39E042505
59550000 rc 0ssse-flw: EmbSe_Rand invoked requesting 16 random bytes
ssse-flw: EmbSe_Rand invoked requesting 16 random bytes
ssse-flw: EmbSe_Rand invoked requesting 16 random bytes
ssse-flw: EmbSe_Rand invoked requesting 32 random bytes
ssse-dbg: EmbSe: EmbSe_RSA_Verify invoked KeyIdent=ff, KeyId=ff, Coeff=0xff
ssse-flw: EmbSe: Not our SE key
ssse-dbg: EmbSe: Verification PASS
ssse-dbg: EmbSe: EmbSe_RSA_Verify invoked KeyIdent=ff, KeyId=ff, Coeff=0xff
ssse-flw: EmbSe: Not our SE key
ssse-dbg: EmbSe: Verification PASS
ssse-dbg: EmbSe: EmbSe_RSA_Verify invoked KeyIdent=ff, KeyId=ff, Coeff=0xff
ssse-flw: EmbSe: Not our SE key
ssse-dbg: EmbSe: Verification PASS
ssse-flw: EmbSe_Rand invoked requesting 32 random bytes
ssse-flw: EmbSe_Compute_Key invoked (ecdh)
ssse-dbg: Requested secret = 32
ssse-dbg: ** nid = 415 **
ssse-flw: No matching key in SE. Invoking OpenSSL API: ECDH_compute_key.
ssse-flw: ECDH_compute_key by OpenSSL PASS
ssse-dbg: Using keyId=0xF0000100
ssse-dbg: shaAlgo: 502
ssse-flw: SSS based sign (keyId=0xF0000100, dgstLen=32)
ssse-flw: SSS based sign called successfully (sigDERLen=70)
ssse-flw: EmbSe_ECDSA_Do_Sign success.
ssse-flw: EmbSe_Rand invoked requesting 8 random bytes
Message with token value 1 delivery confirmed
Message #0 arrived
     topic: $dps/registrations/res/200/?$rid=1
   message: {"operationId":"4.cb935f0666af04f3.1e1e50cf-e009-4253-ad6d-d57eb48a0
685","status":"assigning"}
Device State is assigning
Message with token value 3 delivery confirmed
Message #1 arrived
     topic: $dps/registrations/res/200/?$rid=2
   message: {"operationId":"4.cb935f0666af04f3.1e1e50cf-e009-4253-ad6d-d57eb48a0
685","status":"assigning","registrationState":{"registrationId":"CloudConn0-ECC-
04005001C02EDCE5C8C39E04250559550000","status":"assigning"}}
Device State is assigning
Message with token value 4 delivery confirmed
Message #2 arrived
     topic: $dps/registrations/res/200/?$rid=2
   message: {"operationId":"4.cb935f0666af04f3.1e1e50cf-e009-4253-ad6d-d57eb48a0
685","status":"assigning","registrationState":{"registrationId":"CloudConn0-ECC-
04005001C02EDCE5C8C39E04250559550000","status":"assigning"}}
Device State is assigning
Message with token value 5 delivery confirmed
Message #3 arrived
     topic: $dps/registrations/res/200/?$rid=2
   message: {"operationId":"4.cb935f0666af04f3.1e1e50cf-e009-4253-ad6d-d57eb48a0
685","status":"assigned","registrationState":{"x509":{},"registrationId":"CloudC
onn0-ECC-04005001C02EDCE5C8C39E04250559550000","createdDateTimeUtc":"2019-11-21T
15:15:37.1530301Z","assignedHub":"MyIoTHubName.azure-devices.net","deviceId":"Cl
oudConn0-ECC-04005001C02EDCE5C8C39E04250559550000","status":"assigned","substatu
s":"initialAssignment","lastUpdatedDateTimeUtc":"2019-11-21T15:15:37.437534Z","e
tag":"ImEyMDAyMDJjLTAwMDAtMGQwMC0wMDAwLTVkZDZhYTE5MDAwMCI="}}
Device State is now ASSIGNED
Publish done with error code : 0
root@imx6ulevk:~/se050_mw_v02.11.01/simw-top/demos/linux/azure# 
```

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



You can view and update the enrollment details for an individual enrollment or remove the registration record for a previously provisioned device

Registration Status

Status: assigned

Assigned hub: MyIoTHubName.azure-devices.net

Device ID: CloudConn0-ECC-04005001C02EDCE5C8C39E04250559550000

Last assigned: Thu Nov 21 2019 16:15:37 GMT+0100 (Romance Standard Time)

Authentication Type

Mechanism: X.509

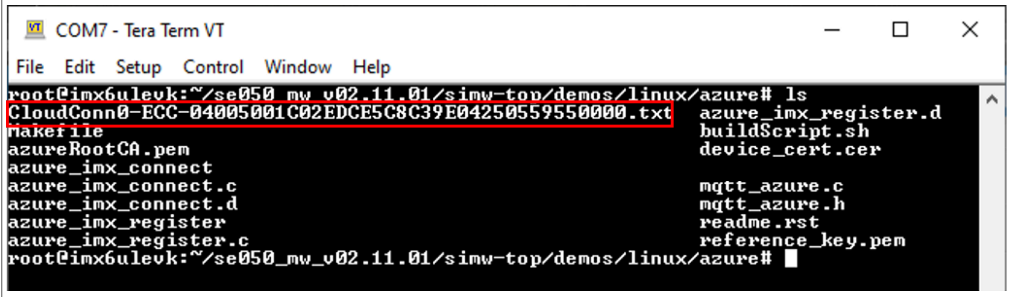
Primary Certificate Thumbprint

43CD6477C2BABFB9CAA0E5C34F809D9428427282

Primary Certificate .pem or .cer file

Figure 28. Device registered in Azure platform.

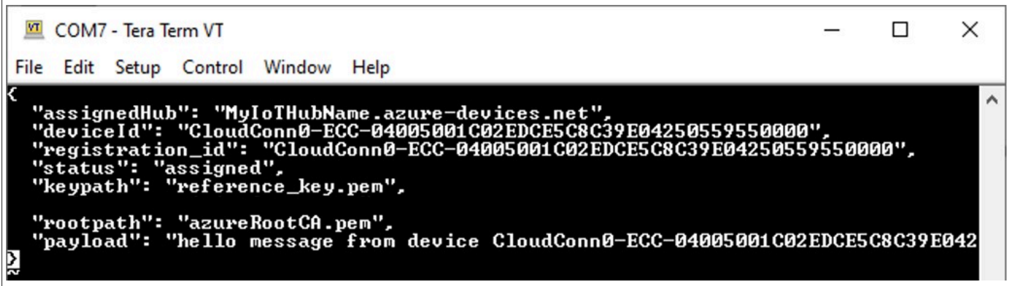
5. If you run the `ls` command, you will observe a new *CloudConn.txt* file in the directory as shown in Figure 29. This file will be used in Section 3.9 for the connection of the device.



```
root@imx6ulevk:~/se050_mw_v02.11.01/simw-top/demos/linux/azure# ls
CloudConn0-ECC-04005001C02EDCE5C8C39E04250559550000.txt  azure_imx_register.d
makefile                                                 buildScript.sh
azureRootCA.pem                                          device_cert.cer
azure_imx_connect
azure_imx_connect.c                                      mqtt_azure.c
azure_imx_connect.d                                      mqtt_azure.h
azure_imx_register                                       readme.rst
azure_imx_register.c                                     reference_key.pem
root@imx6ulevk:~/se050_mw_v02.11.01/simw-top/demos/linux/azure#
```

Figure 29. CloudConn .txt file creation.

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



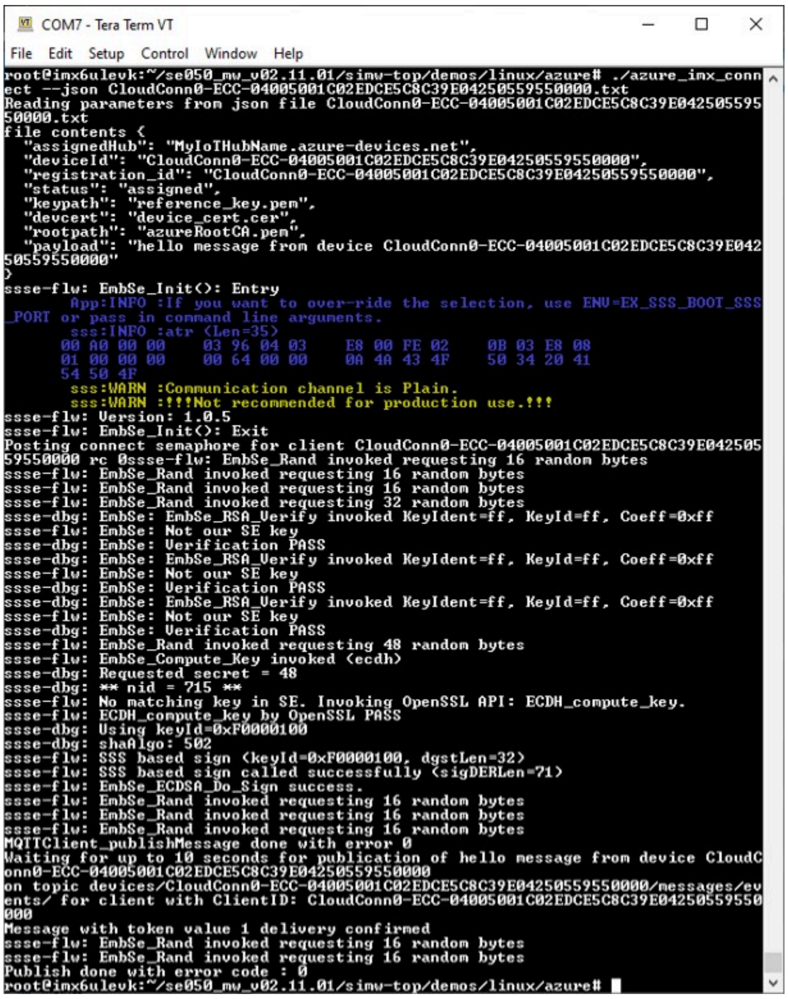
```
{
  "assignedHub": "MyIoTHubName.azure-devices.net",
  "deviceId": "CloudConn0-ECC-04005001C02EDCE5C8C39E04250559550000",
  "registration_id": "CloudConn0-ECC-04005001C02EDCE5C8C39E04250559550000",
  "status": "assigned",
  "keypath": "reference_key.pem",

  "rootpath": "azureRootCA.pem",
  "payload": "hello message from device CloudConn0-ECC-04005001C02EDCE5C8C39E042
}
~
```

Figure 30. CloudCon .txt file contents

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:

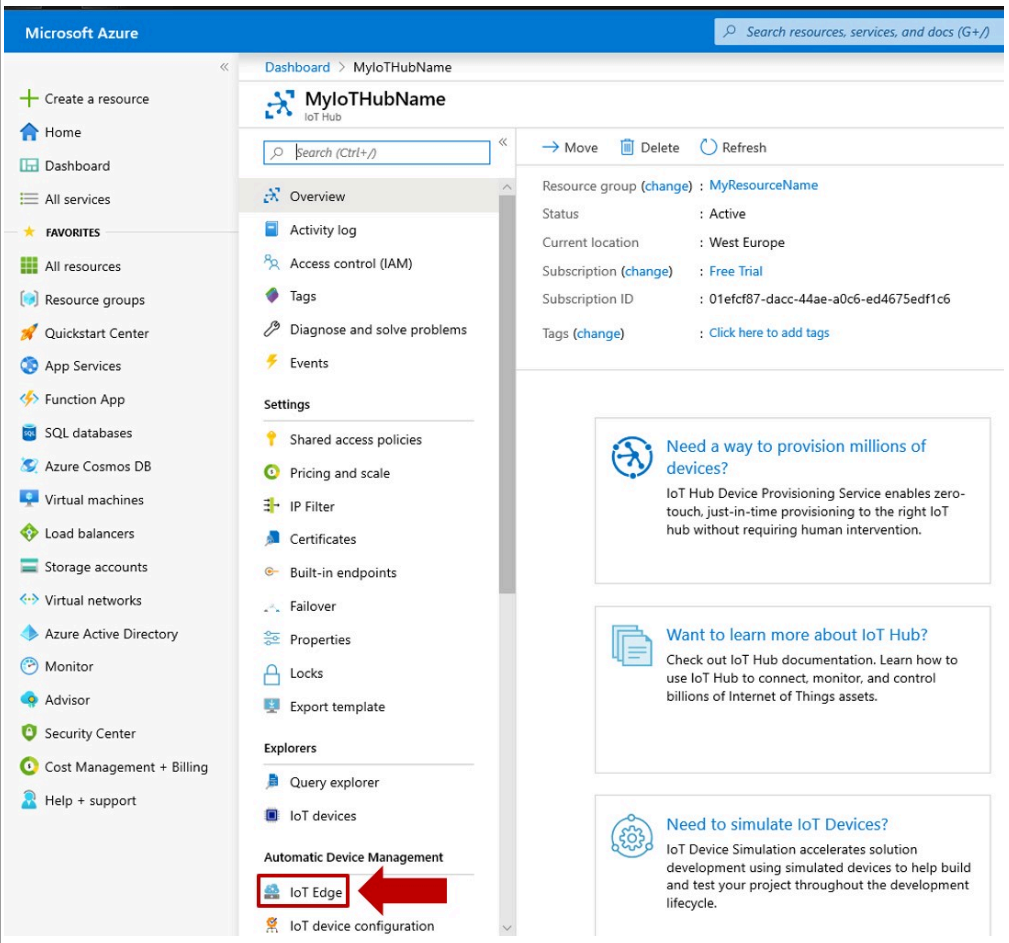
1. Execute the `azure_imx_connect` script.
   Send > `./azure_imx_connect --json CloudConn0-ECC-04005001C02EDCE5C8C39E04250559550000.txt`.



```
root@imx6ulevk:~/se050_mw_v02.11.01/simw-top/demos/linux/azure# ./azure_imx_conn
ect --json CloudConn0-ECC-04005001C02EDCE5C8C39E04250559550000.txt
Reading parameters from json file CloudConn0-ECC-04005001C02EDCE5C8C39E042505595
50000.txt
file contents {
  "assignedHub": "MyIoTHubName.azure-devices.net",
  "deviceId": "CloudConn0-ECC-04005001C02EDCE5C8C39E04250559550000",
  "registration_id": "CloudConn0-ECC-04005001C02EDCE5C8C39E04250559550000",
  "status": "assigned",
  "keypath": "reference_key.pem",
  "devcert": "device_cert.cer",
  "rootpath": "azureRootCA.pem",
  "payload": "hello message from device CloudConn0-ECC-04005001C02EDCE5C8C39E042
50559550000"
}
ssse-flw: EmbSe_Init(): Entry
       App:INFO :If you want to over-ride the selection, use ENV=EX_SSS_BOOT_SSS
_PORT or pass in command line arguments.
       sss:INFO :atr (Len=35)
      00 A0 00 00    03 96 04 03    E8 00 FE 02    0B 03 E8 08
      01 00 00 00    00 64 00 00    0A 4A 43 4F    50 34 20 41
      54 50 4F
       sss:WARN :Communication channel is Plain.
       sss:WARN :!!!Not recommended for production use.!!!
ssse-flw: Version: 1.0.5
ssse-flw: EmbSe_Init(): Exit
Posting connect semaphore for client CloudConn0-ECC-04005001C02EDCE5C8C39E042505
59550000 rc 0ssse-flw: EmbSe_Rand invoked requesting 16 random bytes
ssse-flw: EmbSe_Rand invoked requesting 16 random bytes
ssse-flw: EmbSe_Rand invoked requesting 16 random bytes
ssse-flw: EmbSe_Rand invoked requesting 32 random bytes
ssse-dbg: EmbSe: EmbSe_RSA_Verify invoked KeyIdent=ff, KeyId=ff, Coeff=0xff
ssse-flw: EmbSe: Not our SE key
ssse-dbg: EmbSe: Verification PASS
ssse-dbg: EmbSe: EmbSe_RSA_Verify invoked KeyIdent=ff, KeyId=ff, Coeff=0xff
ssse-flw: EmbSe: Not our SE key
ssse-dbg: EmbSe: Verification PASS
ssse-dbg: EmbSe: EmbSe_RSA_Verify invoked KeyIdent=ff, KeyId=ff, Coeff=0xff
ssse-flw: EmbSe: Not our SE key
ssse-dbg: EmbSe: Verification PASS
ssse-flw: EmbSe_Rand invoked requesting 48 random bytes
ssse-flw: EmbSe_Compute_Key invoked (ecdh)
ssse-dbg: Requested secret = 48
ssse-dbg: ** nid = 715 **
ssse-flw: No matching key in SE. Invoking OpenSSL API: ECDH_compute_key.
ssse-flw: ECDH_compute_key by OpenSSL PASS
ssse-dbg: Using keyId=0xF0000100
ssse-dbg: shaAlgo: 502
ssse-flw: SSS based sign (keyId=0xF0000100, dgstLen=32)
ssse-flw: SSS based sign called successfully (sigDERLen=71)
ssse-flw: EmbSe_ECDSA_Do_Sign success.
ssse-flw: EmbSe_Rand invoked requesting 16 random bytes
ssse-flw: EmbSe_Rand invoked requesting 16 random bytes
ssse-flw: EmbSe_Rand invoked requesting 16 random bytes
MQTTClient_publishMessage done with error 0
Waiting for up to 10 seconds for publication of hello message from device CloudC
onn0-ECC-04005001C02EDCE5C8C39E04250559550000
on topic devices/CloudConn0-ECC-04005001C02EDCE5C8C39E04250559550000/messages/ev
ents/ for client with ClientID: CloudConn0-ECC-04005001C02EDCE5C8C39E04250559550
000
Message with token value 1 delivery confirmed
ssse-flw: EmbSe_Rand invoked requesting 16 random bytes
ssse-flw: EmbSe_Rand invoked requesting 16 random bytes
Publish done with error code : 0
root@imx6ulevk:~/se050_mw_v02.11.01/simw-top/demos/linux/azure# 
```

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 .



Resource group (change) : MyResourceName
Status : Active
Current location : West Europe
Subscription (change) : Free Trial
Subscription ID : 01efcf87-dacc-44ae-a0c6-ed4675edf1c6
Tags (change) : Click here to add tags

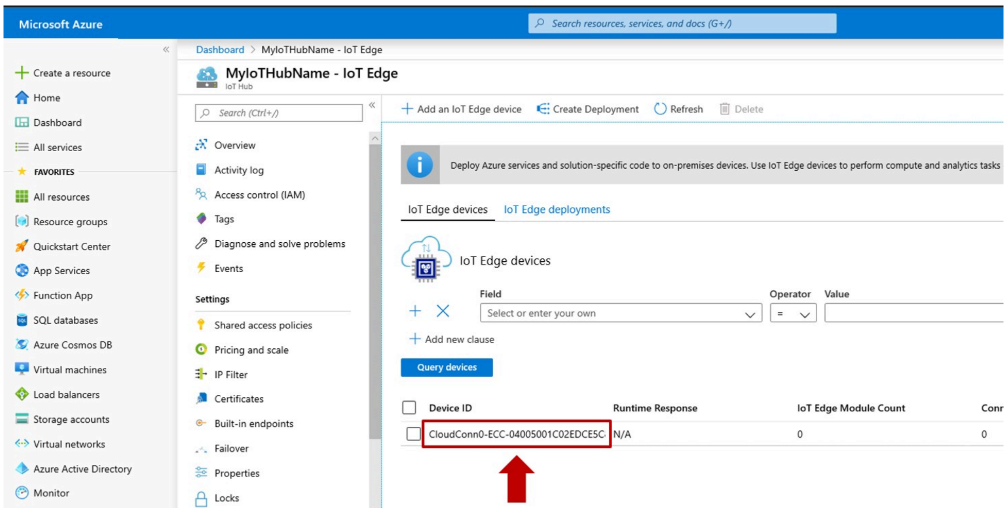
Need a way to provision millions of devices?
IoT Hub Device Provisioning Service enables zero-touch, just-in-time provisioning to the right IoT hub without requiring human intervention.

Want to learn more about IoT Hub?
Check out IoT Hub documentation. Learn how to use IoT Hub to connect, monitor, and control billions of Internet of Things assets.

Need to simulate IoT Devices?
IoT Device Simulation accelerates solution development using simulated devices to help build and test your project throughout the development lifecycle.

Figure 32. Select IoT Edge in IoT Hub overview.

3. The device should appear in the list of IoT Edge devices, as shown in Figure 33.



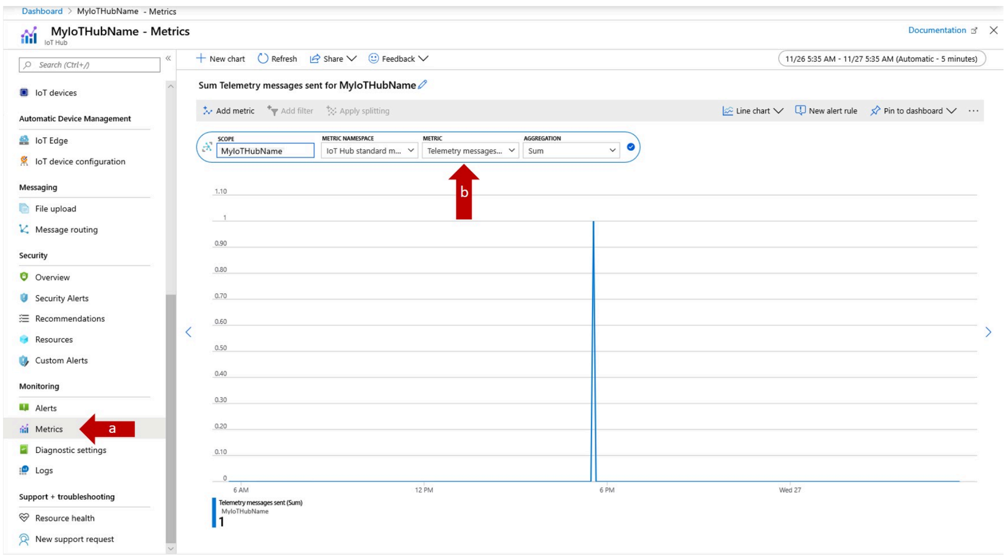
Deploy Azure services and solution-specific code to on-premises devices. Use IoT Edge devices to perform compute and analytics tasks

IoT Edge devices

Device ID	Runtime Response	IoT Edge Module Count	Conn
CloudConn0-ECC-04005001C02EDCE5C	N/A	0	0

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 .



11/26 5:35 AM - 11/27 5:35 AM (Automatic - 5 minutes)

Sum Telemetry messages sent for MyloTHubName

SCOPE: MyloTHubName
METRIC NAMESPACE: IoT Hub standard m...
METRIC: Telemetry messages...
AGGREGATION: Sum

Telemetry messages sent (Sum)
MyloTHubName
1

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 4. OM-SE050ARD development kit details

Part number	12NC	Content	Picture
OM-SE050ARD	935383282598	EdgeLock SE050 development board	

2. FRDM-K64F board:

Table 5. FRDM-K64F details

Part number	12NC	Content	Picture
FRDM-64F	935326293598	Freedom development platform for Kinetis K64, K63 and K24 MCUs	

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

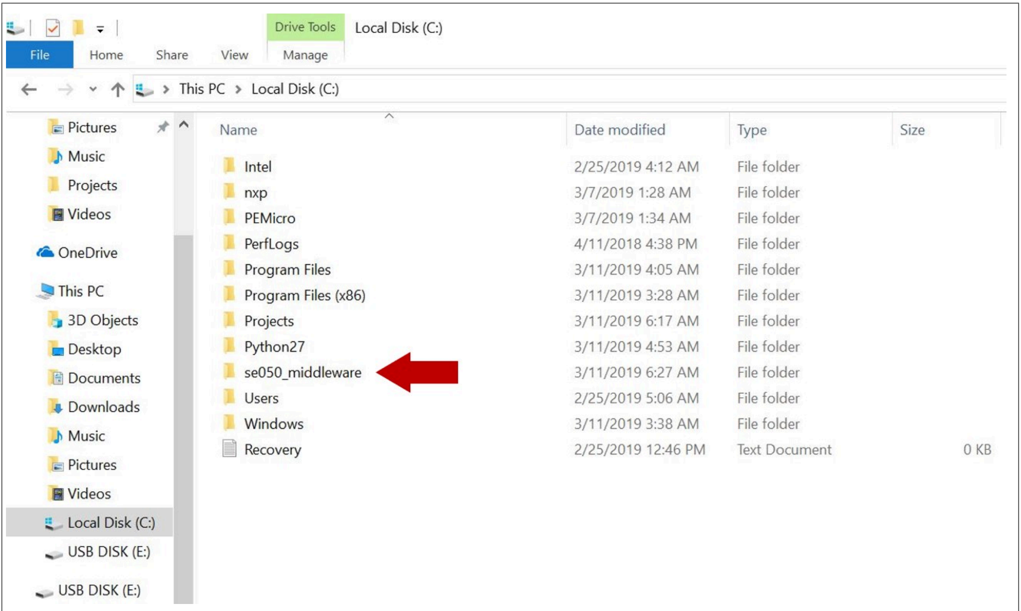
This section explains how to generate the credentials for the EdgeLock SE05x using the key provisioning script included in EdgeLock SE05x Plug & Trust middleware and a FRDM-K64F board as a host platform. These credentials are required for the device onboarding into Azure IoT Hub.

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:

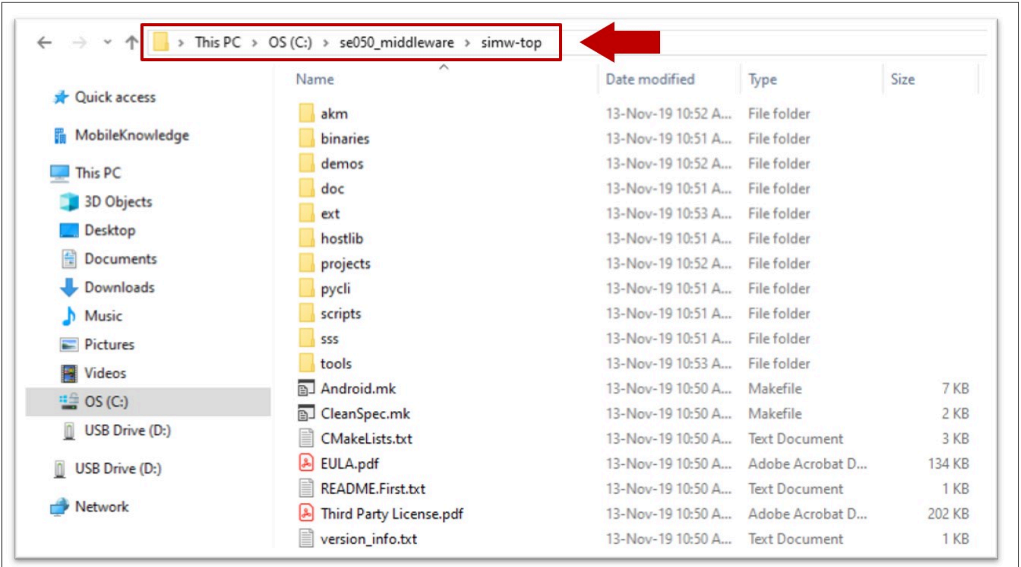
1. Download EdgeLock SE05x Plug & Trust middleware from www.nxp.com/se050.
2. Create a folder called ***se050_middleware*** in C: directory as shown in Figure 35:



Name	Date modified	Type	Size
Intel	2/25/2019 4:12 AM	File folder	
nxp	3/7/2019 1:28 AM	File folder	
PEMicro	3/7/2019 1:34 AM	File folder	
PerfLogs	4/11/2018 4:38 PM	File folder	
Program Files	3/11/2019 4:05 AM	File folder	
Program Files (x86)	3/11/2019 3:28 AM	File folder	
Projects	3/11/2019 6:17 AM	File folder	
Python27	3/11/2019 4:53 AM	File folder	
se050_middleware	3/11/2019 6:27 AM	File folder	
Users	2/25/2019 5:06 AM	File folder	
Windows	3/11/2019 3:38 AM	File folder	
Recovery	2/25/2019 12:46 PM	Text Document	0 KB

Figure 35. Create se050_middleware folder

3. Create a folder calle ***simw-top*** inside the *se050_middleware*. Unzip SE050 middleware inside the ***simw-top*** folder. The unzipped package should look as shown in Figure 36:



This PC > OS (C:) > se050_middleware > simw-top

Name	Date modified	Type	Size
akm	13-Nov-19 10:52 A...	File folder	
binaries	13-Nov-19 10:51 A...	File folder	
demos	13-Nov-19 10:52 A...	File folder	
doc	13-Nov-19 10:51 A...	File folder	
ext	13-Nov-19 10:53 A...	File folder	
hostlib	13-Nov-19 10:51 A...	File folder	
projects	13-Nov-19 10:52 A...	File folder	
pycli	13-Nov-19 10:51 A...	File folder	
scripts	13-Nov-19 10:51 A...	File folder	
sss	13-Nov-19 10:51 A...	File folder	
tools	13-Nov-19 10:53 A...	File folder	
Android.mk	13-Nov-19 10:50 A...	Makefile	7 KB
CleanSpec.mk	13-Nov-19 10:50 A...	Makefile	2 KB
CMakeLists.txt	13-Nov-19 10:50 A...	Text Document	3 KB
EULA.pdf	13-Nov-19 10:50 A...	Adobe Acrobat D...	134 KB
README.First.txt	13-Nov-19 10:50 A...	Text Document	1 KB
Third Party License.pdf	13-Nov-19 10:50 A...	Adobe Acrobat D...	202 KB
version_info.txt	13-Nov-19 10:50 A...	Text Document	1 KB

Figure 36. Unzip se050 middleware

Note: *It is recommended to keep* `simw-top` *with the* ***<u>shortest</u>*** *path possible and* ***<u>without spaces</u>*** *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:

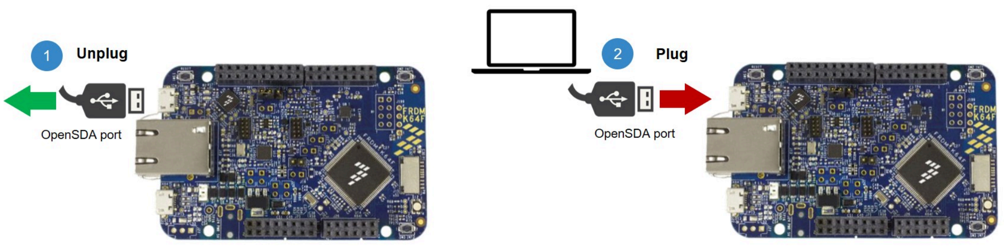
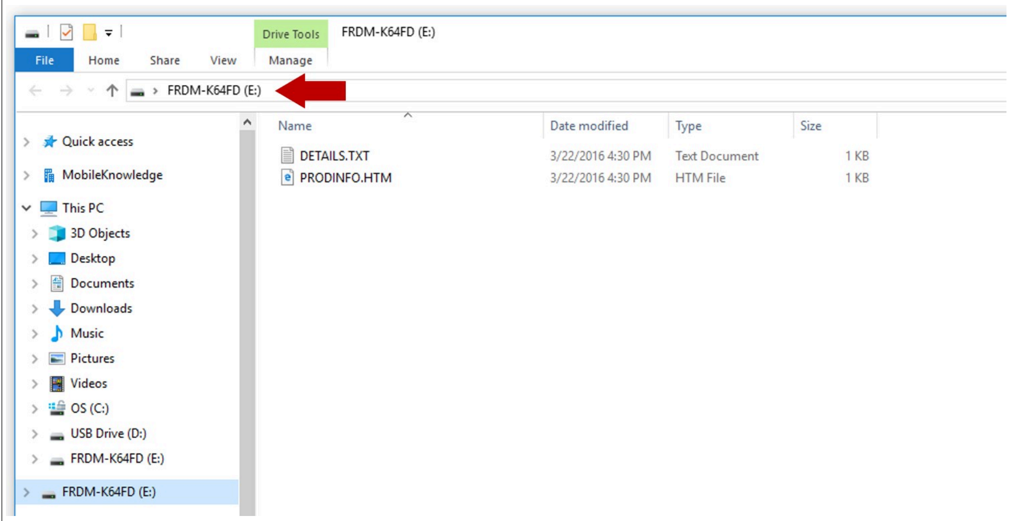


Figure 37. Unplug and plug OpenSDA port

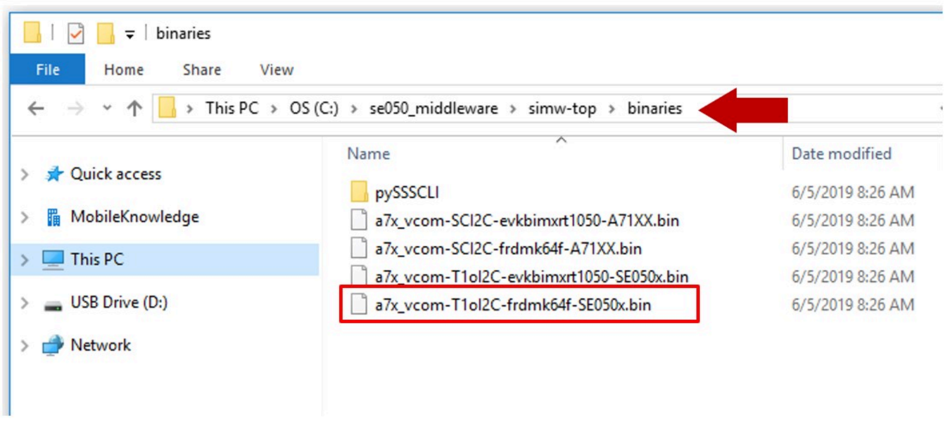
2. When you plug the board, your laptop should recognize the board as an external drive as shown in Figure 38:



Name	Date modified	Type	Size
DETAILS.TXT	3/22/2016 4:30 PM	Text Document	1 KB
PRODINFO.HTM	3/22/2016 4:30 PM	HTM File	1 KB

Figure 38. FRDM-K64F drive

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:



Name	Date modified
pySSSCLI	6/5/2019 8:26 AM
a7x_vcom-SCI2C-evkbimxrt1050-A71XX.bin	6/5/2019 8:26 AM
a7x_vcom-SCI2C-frdmk64f-A71XX.bin	6/5/2019 8:26 AM
a7x_vcom-T1oI2C-evkbimxrt1050-SE050x.bin	6/5/2019 8:26 AM
a7x_vcom-T1oI2C-frdmk64f-SE050x.bin	6/5/2019 8:26 AM

Figure 39. VCOM binary folder

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:

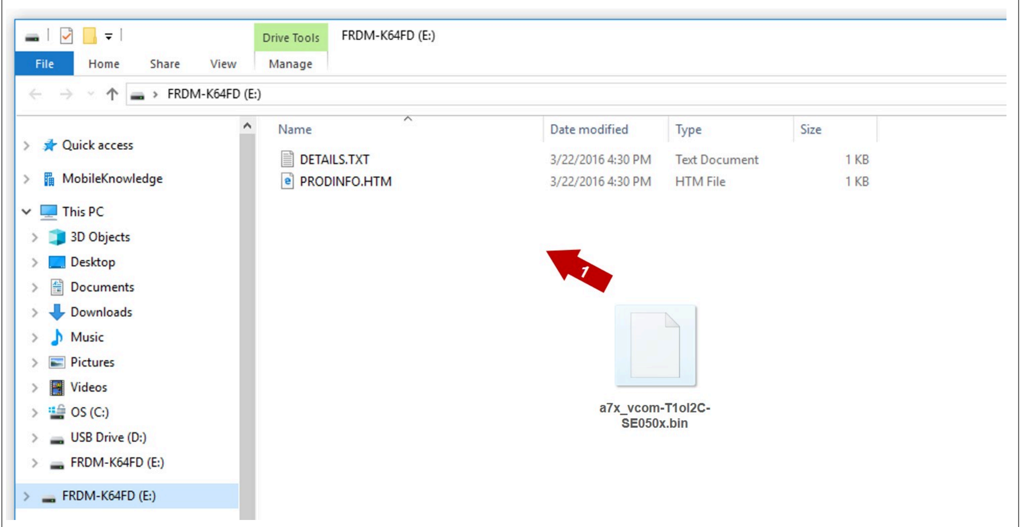


Figure 40. Drag and drop VCOM binary

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.

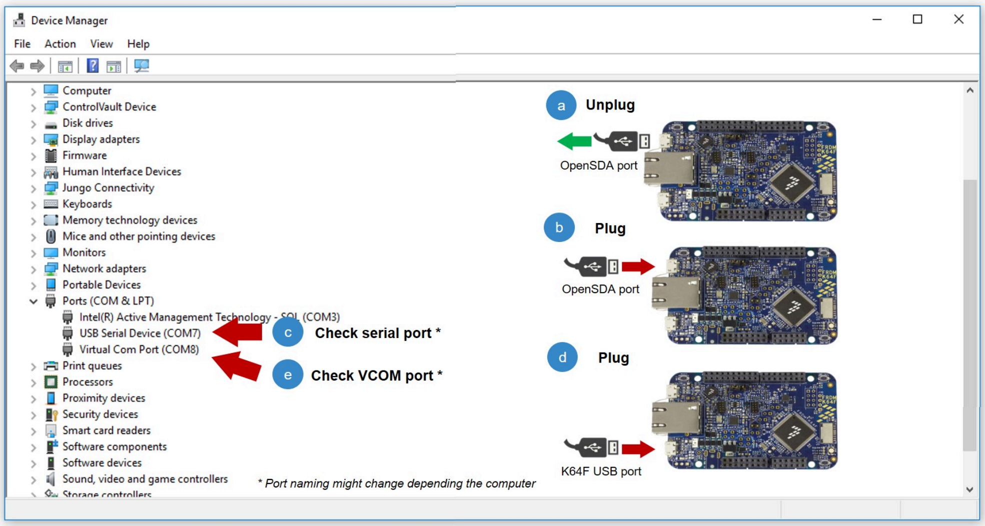


Figure 41. Check VCOM and serial ports

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

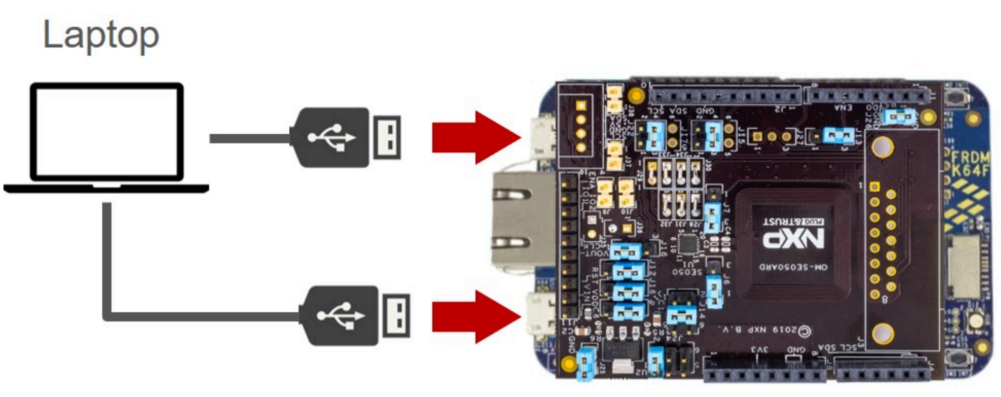
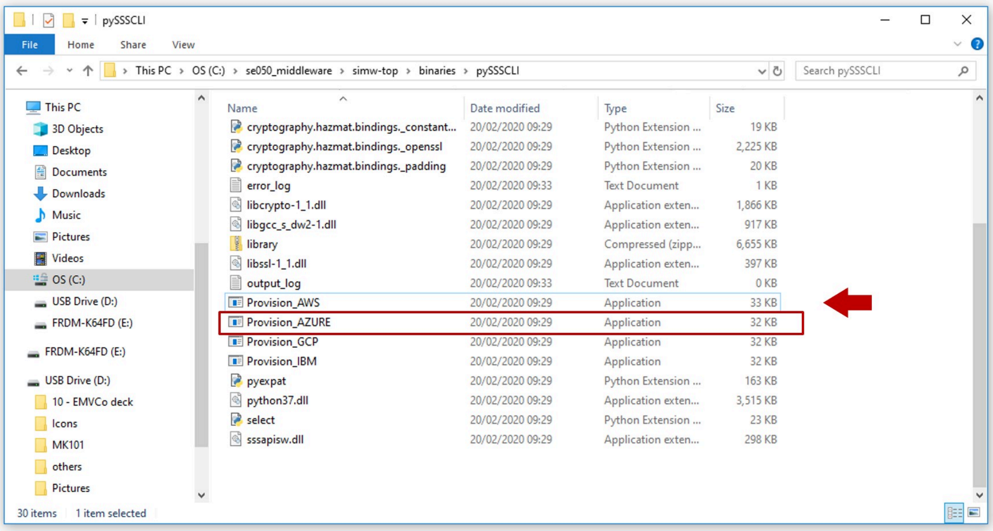


Figure 42. Connect boards

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



pySSSCLI

File | Home | Share | View

This PC › OS (C:) › se050_middleware › simw-top › binaries › pySSSCLI

Search pySSSCLI

- This PC
  - 3D Objects
  - Desktop
  - Documents
  - Downloads
  - Music
  - Pictures
  - Videos
  - OS (C:)
  - USB Drive (D:)
  - FRDM-K64FD (E:)
- FRDM-K64FD (E:)
- USB Drive (D:)
  - 10 - EMVCo deck
  - Icons
  - MK101
  - others
  - Pictures

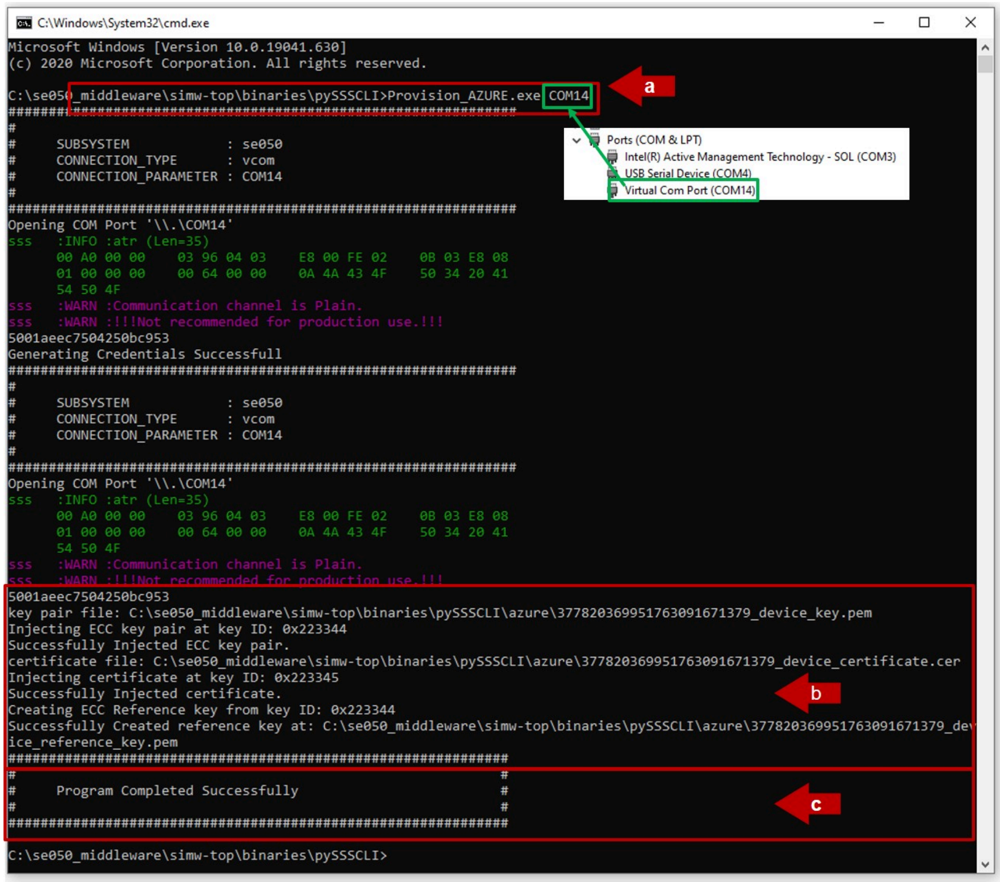
Name	Date modified	Type	Size
cryptography.hazmat.bindings._constant...	20/02/2020 09:29	Python Extension ...	19 KB
cryptography.hazmat.bindings._openssl	20/02/2020 09:29	Python Extension ...	2,225 KB
cryptography.hazmat.bindings._padding	20/02/2020 09:29	Python Extension ...	20 KB
error_log	20/02/2020 09:33	Text Document	1 KB
libcrypto-1_1.dll	20/02/2020 09:29	Application exten...	1,866 KB
libgcc_s_dw2-1.dll	20/02/2020 09:29	Application exten...	917 KB
library	20/02/2020 09:29	Compressed (zipp...	6,655 KB
libssl-1_1.dll	20/02/2020 09:29	Application exten...	397 KB
output_log	20/02/2020 09:33	Text Document	0 KB
Provision_AWS	20/02/2020 09:29	Application	33 KB
Provision_AZURE	20/02/2020 09:29	Application	32 KB
Provision_GCP	20/02/2020 09:29	Application	32 KB
Provision_IBM	20/02/2020 09:29	Application	32 KB
pyexpat	20/02/2020 09:29	Python Extension ...	163 KB
python37.dll	20/02/2020 09:29	Application exten...	3,515 KB
select	20/02/2020 09:29	Python Extension ...	23 KB
sssapisw.dll	20/02/2020 09:29	Application exten...	298 KB

30 items | 1 item selected

Figure 43. Find Provision_AZURE.exe file in your EdgeLock SE05x Plug & Trust middleware package

3. Open a command prompt

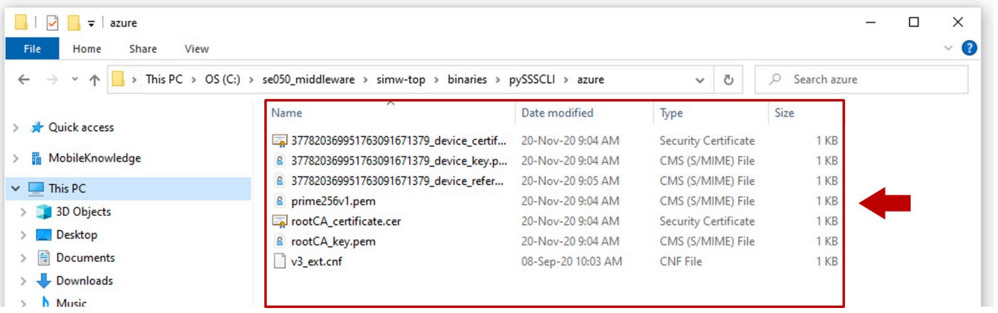
4. Use the `Provision_AZURE.exe` executable to generate and inject keys into your EdgeLock SE05x. Go to the folder `simw-top\binaries\pySSSCLI` and follow the steps shown in Figure 44:
   a. Run the executable `Provision_AZURE.exe <K64_COM_port_number>`. For that, you also need to indicate the VCOM port 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.


C:\Windows\System32\cmd.exe
a
Ports (COM & LPT)
Intel(R) Active Management Technology - SOL (COM3)
USB Serial Device (COM4)
Virtual Com Port (COM14)
b
c

```
Microsoft Windows [Version 10.0.19041.630]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\se050_middleware\simw-top\binaries\pySSSCLI>Provision_AZURE.exe COM14
###############################################################
#
#     SUBSYSTEM             : se050
#     CONNECTION_TYPE       : vcom
#     CONNECTION_PARAMETER : COM14
#
###############################################################
Opening COM Port '\\.\COM14'
sss   :INFO :atr (Len=35)
      00 A0 00 00    03 96 04 03    E8 00 FE 02    0B 03 E8 08
      01 00 00 00    00 64 00 00    0A 4A 43 4F    50 34 20 41
      54 50 4F
sss   :WARN :Communication channel is Plain.
sss   :WARN :!!!Not recommended for production use.!!!
5001aeec7504250bc953
Generating Credentials Successfull
###############################################################
#
#     SUBSYSTEM             : se050
#     CONNECTION_TYPE       : vcom
#     CONNECTION_PARAMETER : COM14
#
###############################################################
Opening COM Port '\\.\COM14'
sss   :INFO :atr (Len=35)
      00 A0 00 00    03 96 04 03    E8 00 FE 02    0B 03 E8 08
      01 00 00 00    00 64 00 00    0A 4A 43 4F    50 34 20 41
      54 50 4F
sss   :WARN :Communication channel is Plain.
sss   :WARN :!!!Not recommended for production use.!!!
5001aeec7504250bc953
key pair file: C:\se050_middleware\simw-top\binaries\pySSSCLI\azure\377820369951763091671379_device_key.pem
Injecting ECC key pair at key ID: 0x223344
Successfully Injected ECC key pair.
certificate file: C:\se050_middleware\simw-top\binaries\pySSSCLI\azure\377820369951763091671379_device_certificate.cer
Injecting certificate at key ID: 0x223345
Successfully Injected certificate.
Creating ECC Reference key from key ID: 0x223344
Successfully Created reference key at: C:\se050_middleware\simw-top\binaries\pySSSCLI\azure\377820369951763091671379_dev
ice_reference_key.pem
##############################################################
#                                                            #
#     Program Completed Successfully                         #
#                                                            #
##############################################################

C:\se050_middleware\simw-top\binaries\pySSSCLI>
```

Figure 44. Run `Provision_AZURE.exe` executable

5. Go to `simw-top\binaries\pySSCLI\azure` folder and check that the keys appear inside the folder as shown in Figure 45:



Name	Date modified	Type	Size
377820369951763091671379_device_certif...	20-Nov-20 9:04 AM	Security Certificate	1 KB
377820369951763091671379_device_key.p...	20-Nov-20 9:04 AM	CMS (S/MIME) File	1 KB
377820369951763091671379_device_refer...	20-Nov-20 9:05 AM	CMS (S/MIME) File	1 KB
prime256v1.pem	20-Nov-20 9:04 AM	CMS (S/MIME) File	1 KB
rootCA_certificate.cer	20-Nov-20 9:04 AM	Security Certificate	1 KB
rootCA_key.pem	20-Nov-20 9:04 AM	CMS (S/MIME) File	1 KB
v3_ext.cnf	08-Sep-20 10:03 AM	CNF File	1 KB

Figure 45. Generated Azure credentials

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.

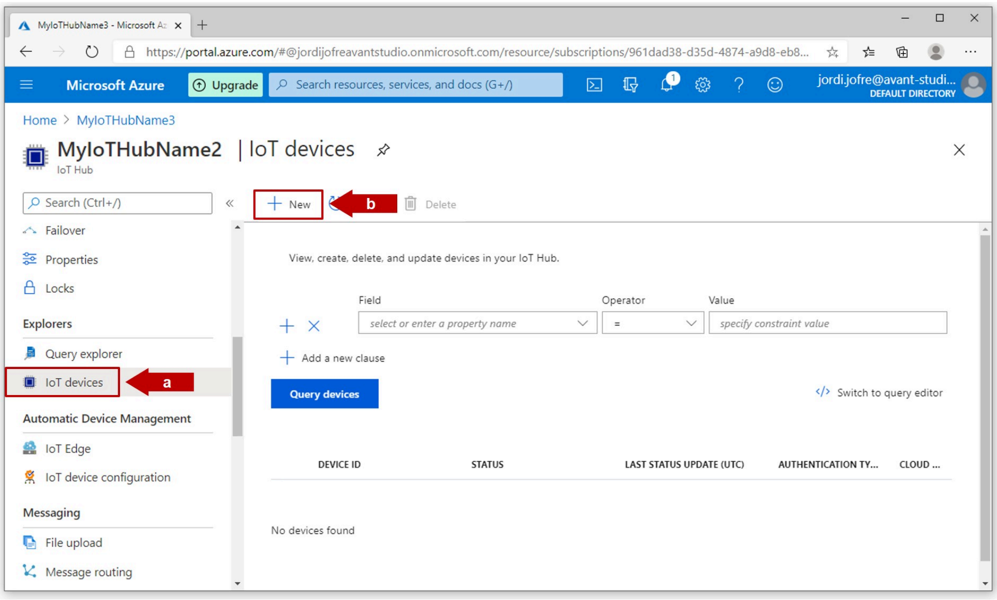
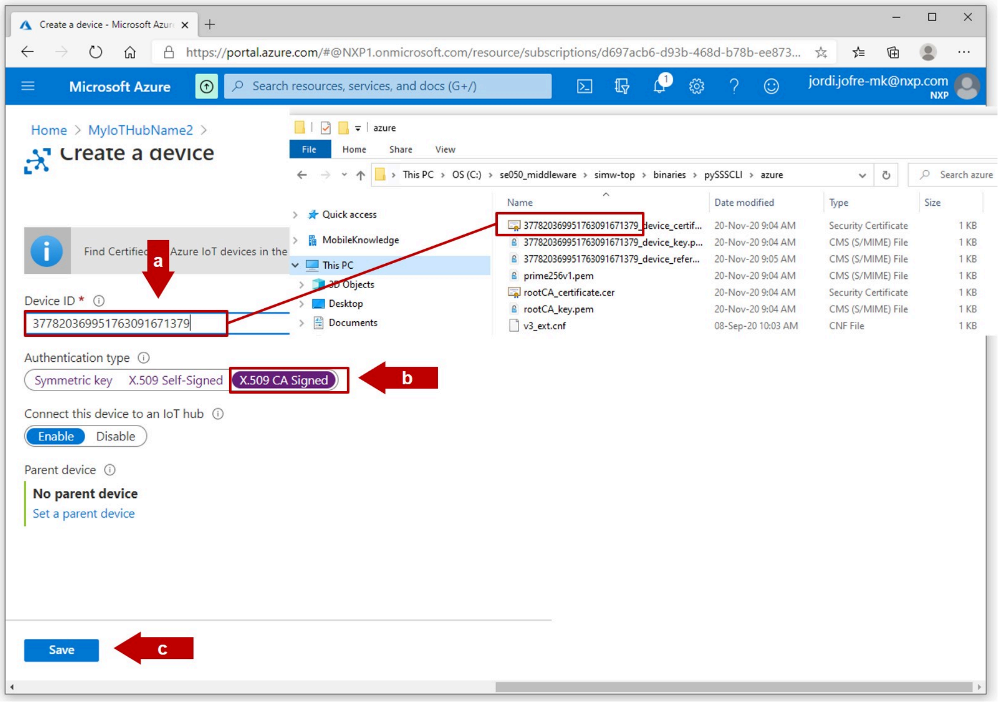


Figure 46. Create a device

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***.


Home > MyIoTHubName2 >

Create a device

Find Certified Azure IoT devices in the

Device ID * 377820369951763091671379

Authentication type: Symmetric key X.509 Self-Signed [✓ X.509 CA Signed]

Connect this device to an IoT hub: [✓ Enable] Disable

Parent device: No parent device

Set a parent device

This PC > OS (C:) > se050_middleware > simw-top > binaries > pySSSCLI > azure

Name	Date modified	Type	Size
377820369951763091671379_device_certif...	20-Nov-20 9:04 AM	Security Certificate	1 KB
377820369951763091671379_device_key.p...	20-Nov-20 9:04 AM	CMS (S/MIME) File	1 KB
377820369951763091671379_device_refer...	20-Nov-20 9:05 AM	CMS (S/MIME) File	1 KB
prime256v1.pem	20-Nov-20 9:04 AM	CMS (S/MIME) File	1 KB
rootCA_certificate.cer	20-Nov-20 9:04 AM	Security Certificate	1 KB
rootCA_key.pem	20-Nov-20 9:04 AM	CMS (S/MIME) File	1 KB
v3_ext.cnf	08-Sep-20 10:03 AM	CNF File	1 KB

Create a device - Microsoft Azure
https://portal.azure.com/#@NXP1.onmicrosoft.com/resource/subscriptions/d697acb6-d93b-468d-b78b-ee873...
Microsoft Azure
Search resources, services, and docs (G+/)
jordi.jofre-mk@nxp.com
NXP
azure
File
Home
Share
View
Search azure
Quick access
MobileKnowledge
This PC
3D Objects
Desktop
Documents
a
b
Save
c

Figure 47. Create a device II

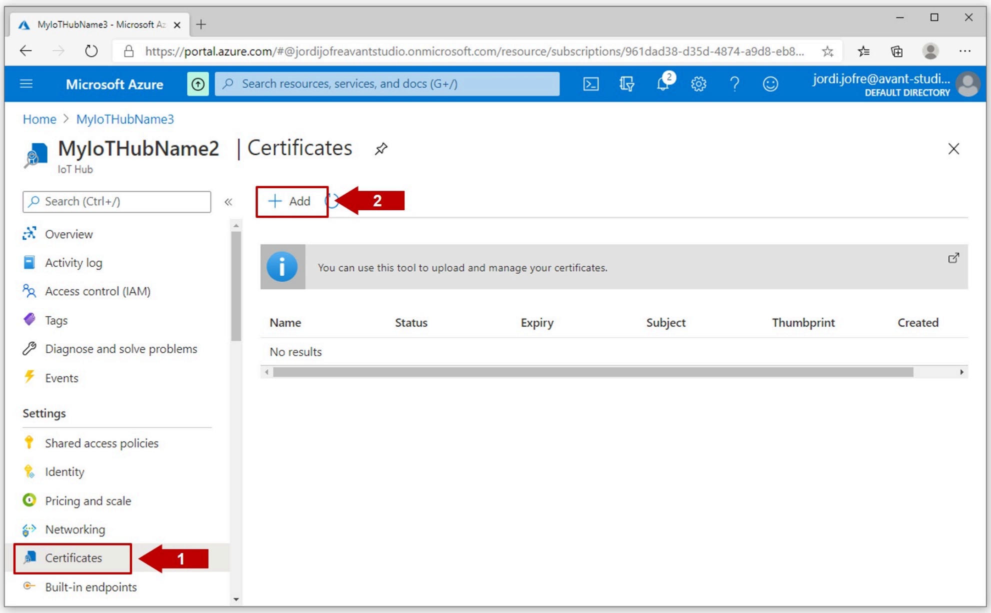
4.3.4 Upload root CA certificate to your Azure IoT Hub

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:


MyloTHubName3 - Microsoft Az
https://portal.azure.com/#@jordijofreavantstudio.onmicrosoft.com/resource/subscriptions/961dad38-d35d-4874-a9d8-eb8...
Microsoft Azure
Search resources, services, and docs (G+/)
jordi.jofre@avant-studi...
DEFAULT DIRECTORY
Home > MyloTHubName3
MyloTHubName2 | Certificates
IoT Hub
Search (Ctrl+/)
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Events
Settings
Shared access policies
Identity
Pricing and scale
Networking
Certificates
Built-in endpoints
+ Add
1
2

You can use this tool to upload and manage your certificates.

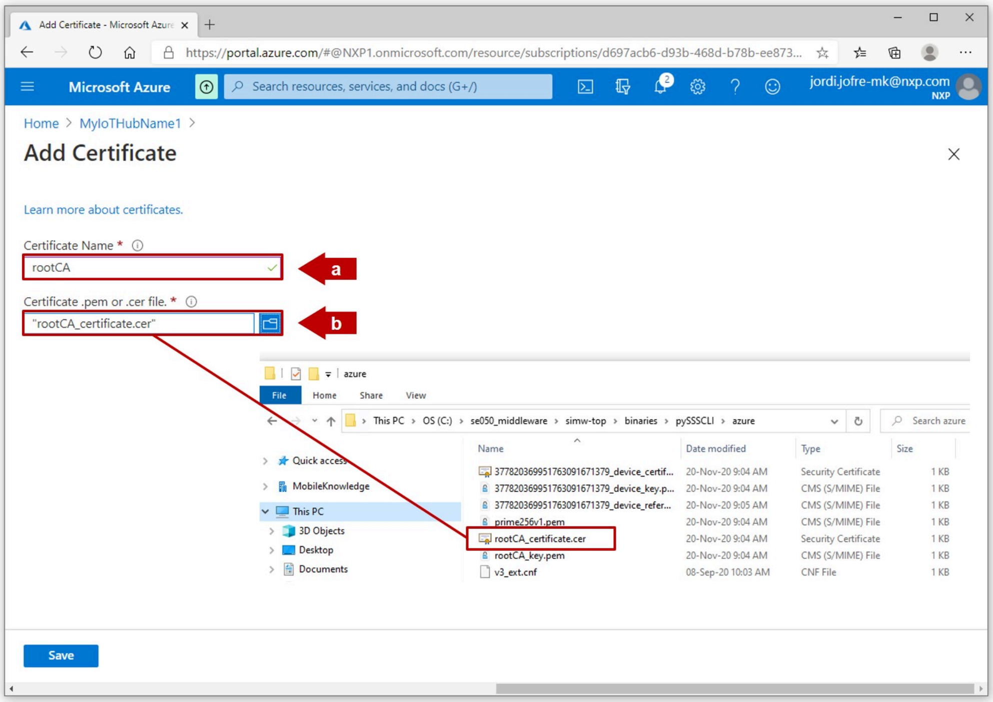
Name	Status	Expiry	Subject	Thumbprint	Created
No results					

Figure 48. Upload a root CA certificate

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

a. As ***Certificate Name***, for instance, use *rootCA_QMC*.

b. Upload the rootCA certificate that we generated in Figure 49.


Add Certificate - Microsoft Azure
https://portal.azure.com/#@NXP1.onmicrosoft.com/resource/subscriptions/d697acb6-d93b-468d-b78b-ee873...
Microsoft Azure
Search resources, services, and docs (G+/)
jordi.jofre-mk@nxp.com
NXP
Home > MyloTHubName1 >
Add Certificate
Learn more about certificates.
a
b
azure
File
Home
Share
View
This PC > OS (C:) > se050_middleware > simw-top > binaries > pySSSCLI > azure
Search azure
Quick access
MobileKnowledge
This PC
3D Objects
Desktop
Documents
Save

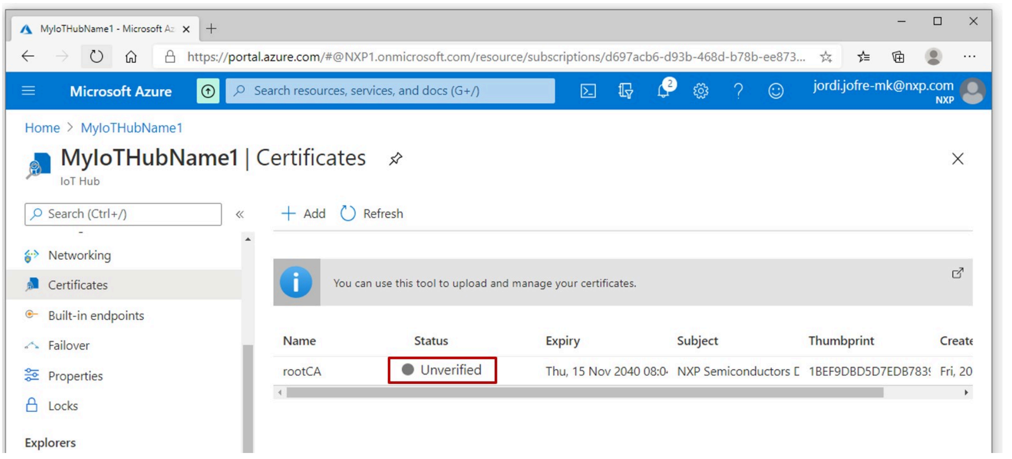
Certificate Name * : rootCA

Certificate .pem or .cer file. * : "rootCA_certificate.cer"

Name	Date modified	Type	Size
377820369951763091671379_device_certif...	20-Nov-20 9:04 AM	Security Certificate	1 KB
377820369951763091671379_device_key.p...	20-Nov-20 9:04 AM	CMS (S/MIME) File	1 KB
377820369951763091671379_device_refer...	20-Nov-20 9:05 AM	CMS (S/MIME) File	1 KB
prime256v1.pem	20-Nov-20 9:04 AM	CMS (S/MIME) File	1 KB
rootCA_certificate.cer	20-Nov-20 9:04 AM	Security Certificate	1 KB
rootCA_key.pem	20-Nov-20 9:04 AM	CMS (S/MIME) File	1 KB
v3_ext.cnf	08-Sep-20 10:03 AM	CNF File	1 KB

Figure 49. Upload a root CA certificate II

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.

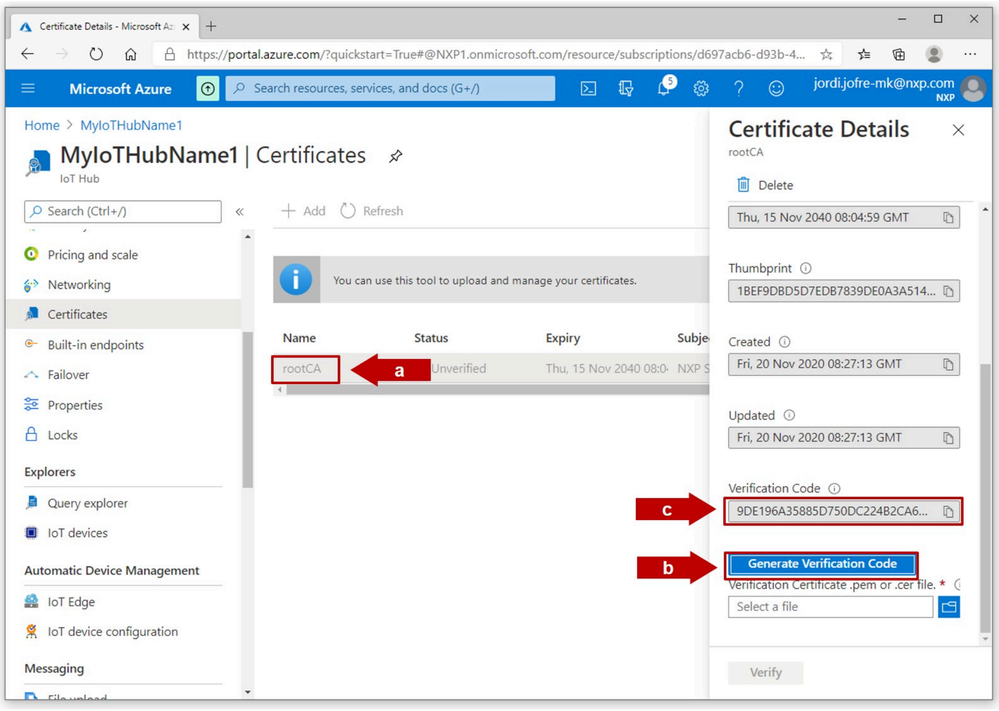

MyIoTHubName1 - Microsoft Az
https://portal.azure.com/#@NXP1.onmicrosoft.com/resource/subscriptions/d697acb6-d93b-468d-b78b-ee873...
Microsoft Azure
Search resources, services, and docs (G+/)
jordi.jofre-mk@nxp.com
NXP
Home > MyIoTHubName1
MyIoTHubName1 | Certificates
IoT Hub
Search (Ctrl+/)
Networking
Certificates
Built-in endpoints
Failover
Properties
Locks
Explorers
+ Add
Refresh

You can use this tool to upload and manage your certificates.

Name	Status	Expiry	Subject	Thumbprint	Create
rootCA	Unverified	Thu, 15 Nov 2040 08:0	NXP Semiconductors [	1BEF9DBD5D7EDB783	Fri, 20

Figure 50. Root CA certificate unverified status

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.



Expiry: Thu, 15 Nov 2040 08:04:59 GMT

Thumbprint: 1BEF9DBD5D7EDB7839DE0A3A514...

Created: Fri, 20 Nov 2020 08:27:13 GMT

Updated: Fri, 20 Nov 2020 08:27:13 GMT

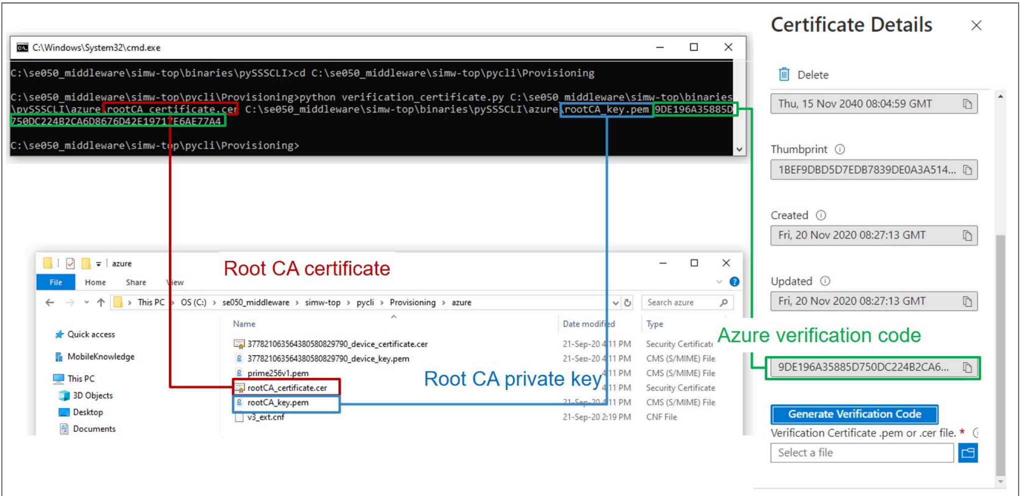
Verification Code: 9DE196A35885D750DC224B2CA6...

Verification Certificate .pem or .cer file. *

Select a file

Figure 51. Obtain root CA certificate verification code

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:



```
C:\se050_middleware\simw-top\binaries\pySSSCLI>cd C:\se050_middleware\simw-top\pycli\Provisioning

C:\se050_middleware\simw-top\pycli\Provisioning>python verification_certificate.py C:\se050_middleware\simw-top\binaries
\pySSSCLI\azure\rootCA_certificate.cer C:\se050_middleware\simw-top\binaries\pySSSCLI\azure\rootCA_key.pem 9DE196A35885D
750DC224B2CA6D8676D42E19717E6AE77A4

C:\se050_middleware\simw-top\pycli\Provisioning>
```

Certificate Details

Thu, 15 Nov 2040 08:04:59 GMT

Thumbprint: 1BEF9DBD5D7EDB7839DE0A3A514...

Created: Fri, 20 Nov 2020 08:27:13 GMT

Updated: Fri, 20 Nov 2020 08:27:13 GMT

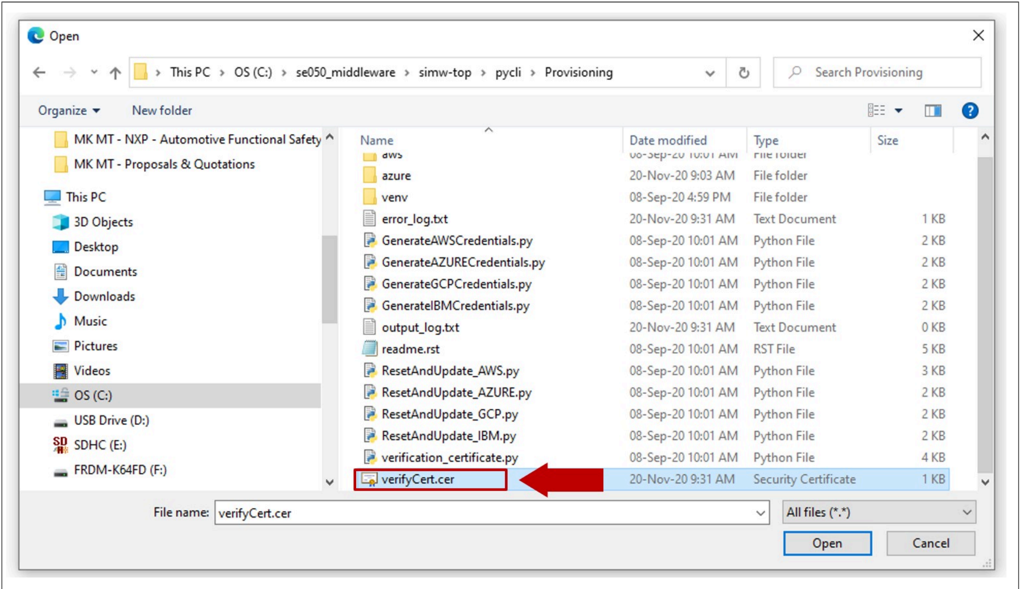
9DE196A35885D750DC224B2CA6...

Verification Certificate .pem or .cer file. *

Delete
Generate Verification Code
Select a file
Root CA certificate
Root CA private key
Azure verification code
This PC › OS (C:) › se050_middleware › simw-top › pycli › Provisioning › azure
377821063564380580829790_device_certificate.cer
377821063564380580829790_device_key.pem
prime256v1.pem
rootCA_certificate.cer
rootCA_key.pem
v3_ext.cnf

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:



Name	Date modified	Type	Size
aws	08-Sep-20 10:01 AM	File folder	
azure	20-Nov-20 9:03 AM	File folder	
venv	08-Sep-20 4:59 PM	File folder	
error_log.txt	20-Nov-20 9:31 AM	Text Document	1 KB
GenerateAWSCredentials.py	08-Sep-20 10:01 AM	Python File	2 KB
GenerateAZURECredentials.py	08-Sep-20 10:01 AM	Python File	2 KB
GenerateGCPCredentials.py	08-Sep-20 10:01 AM	Python File	2 KB
GenerateIBMCredentials.py	08-Sep-20 10:01 AM	Python File	2 KB
output_log.txt	20-Nov-20 9:31 AM	Text Document	0 KB
readme.rst	08-Sep-20 10:01 AM	RST File	5 KB
ResetAndUpdate_AWS.py	08-Sep-20 10:01 AM	Python File	3 KB
ResetAndUpdate_AZURE.py	08-Sep-20 10:01 AM	Python File	2 KB
ResetAndUpdate_GCP.py	08-Sep-20 10:01 AM	Python File	2 KB
ResetAndUpdate_IBM.py	08-Sep-20 10:01 AM	Python File	2 KB
verification_certificate.py	08-Sep-20 10:01 AM	Python File	4 KB
verifyCert.cer	20-Nov-20 9:31 AM	Security Certificate	1 KB

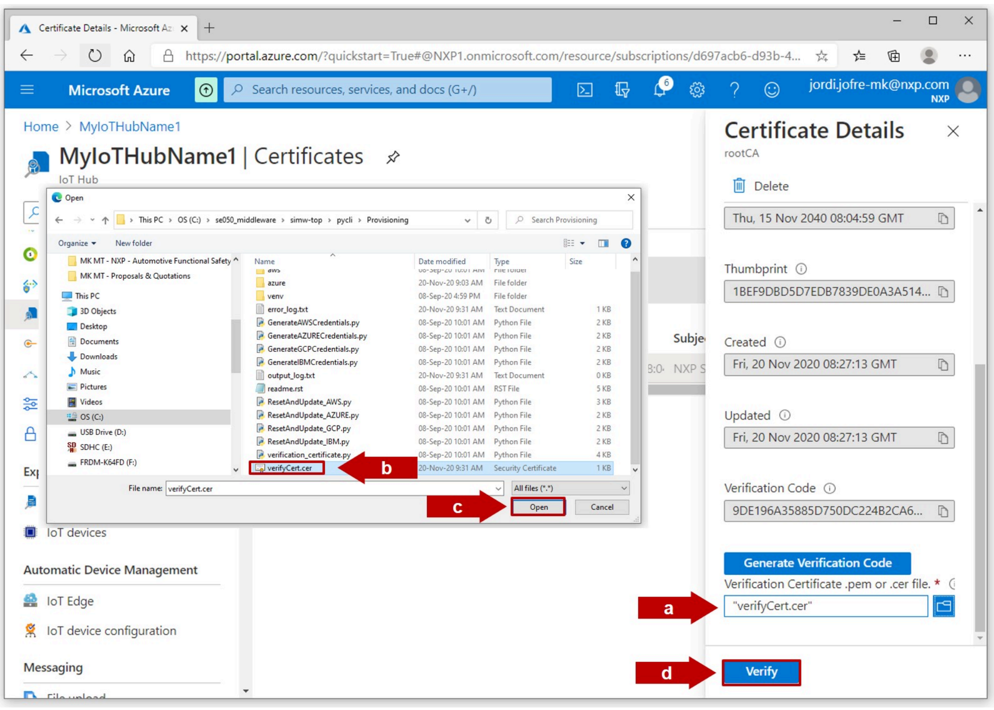
File name: verifyCert.cer

Figure 53. Verification certificated 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***



Certificate Details
rootCA

Thu, 15 Nov 2040 08:04:59 GMT

Thumbprint: 1BEF9DBD5D7EDB7839DE0A3A514...

Created: Fri, 20 Nov 2020 08:27:13 GMT

Updated: Fri, 20 Nov 2020 08:27:13 GMT

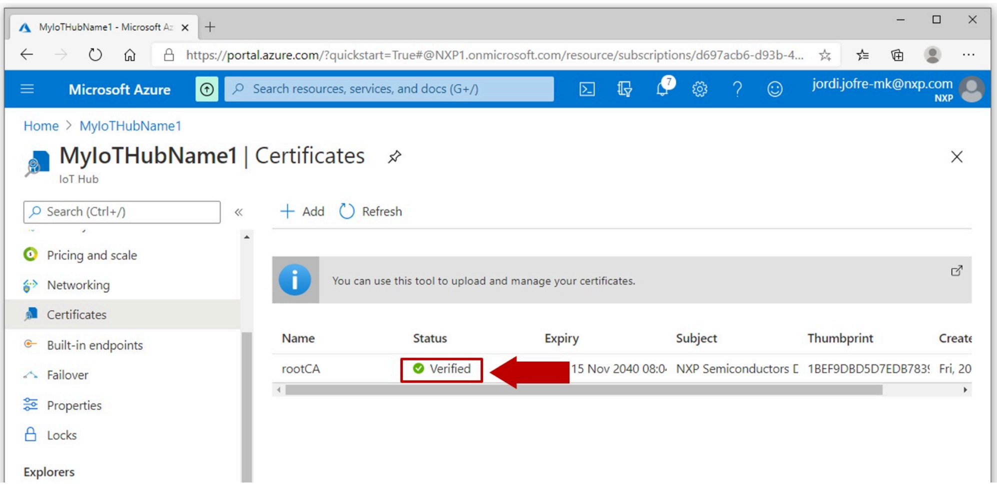
Verification Code: 9DE196A35885D750DC224B2CA6...

Verification Certificate .pem or .cer file. *: "verifyCert.cer"

File name: verifyCert.cer

Figure 54. Root CA certificate verification

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.



You can use this tool to upload and manage your certificates.

Name	Status	Expiry	Subject	Thumbprint	Create
rootCA	Verified	15 Nov 2040 08:0	NXP Semiconductors [	1BEF9DBD5D7EDB783	Fri, 20

Figure 55. Root CA certificate verified

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.

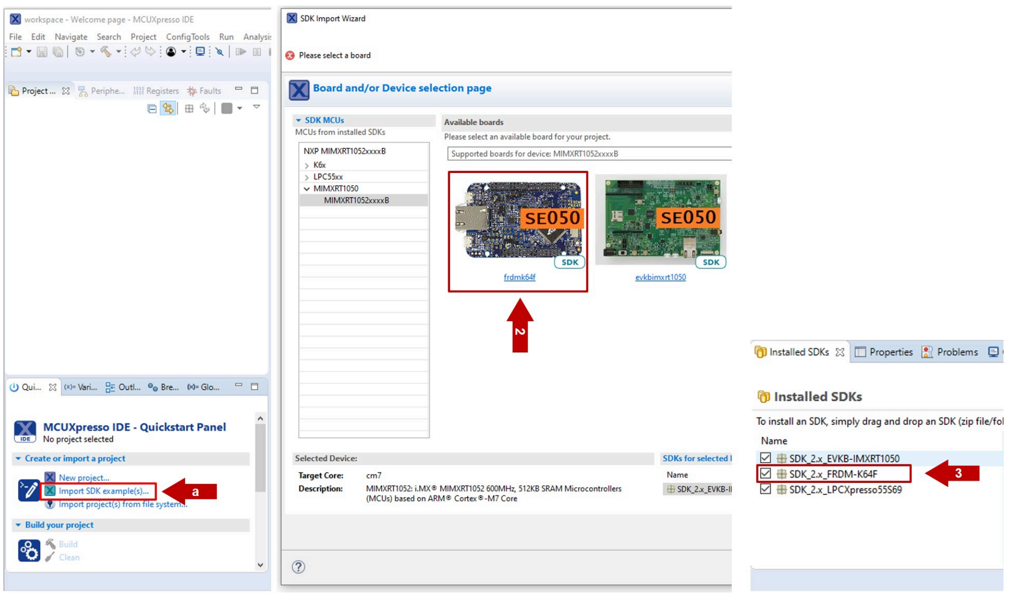


Figure 56. Import FRDM-K64F SDK

Note: *For more detailed instructions on how to install it 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

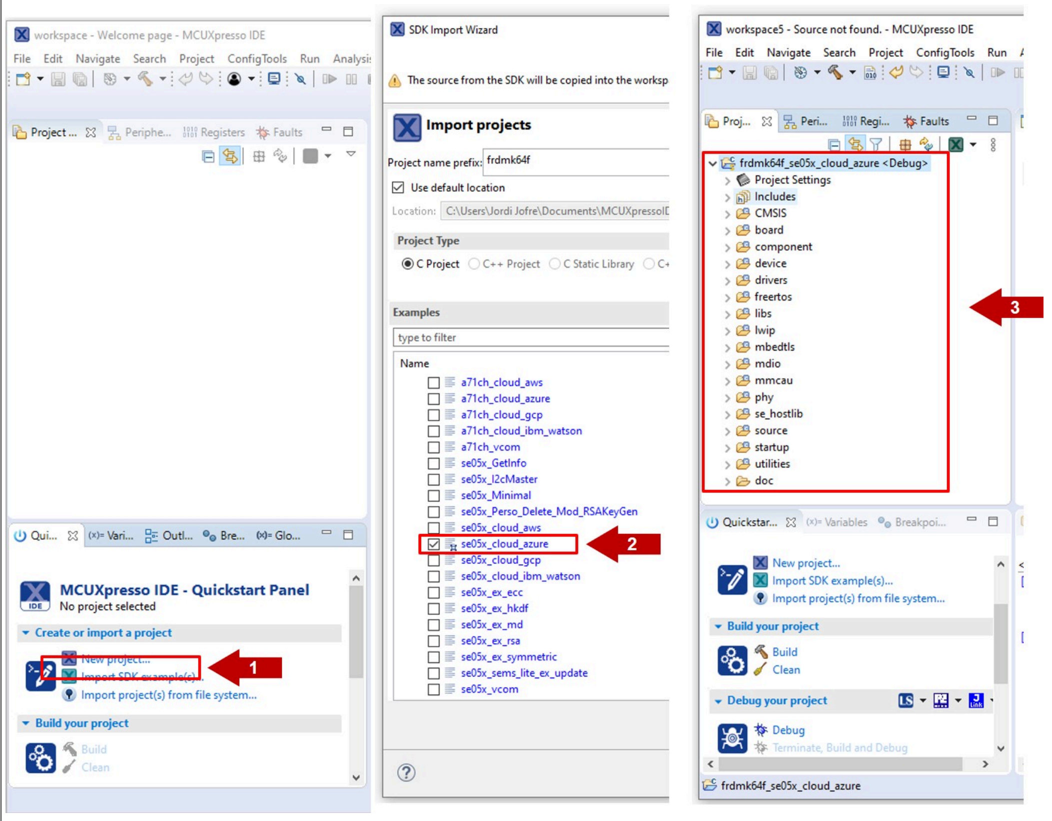
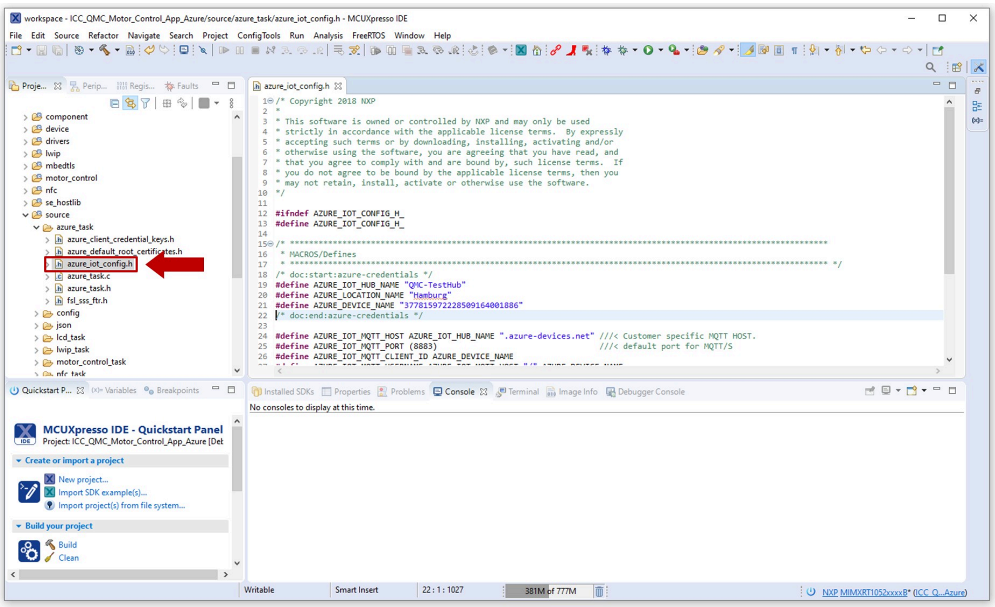


Figure 57. Import Azure project in the workspace

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:

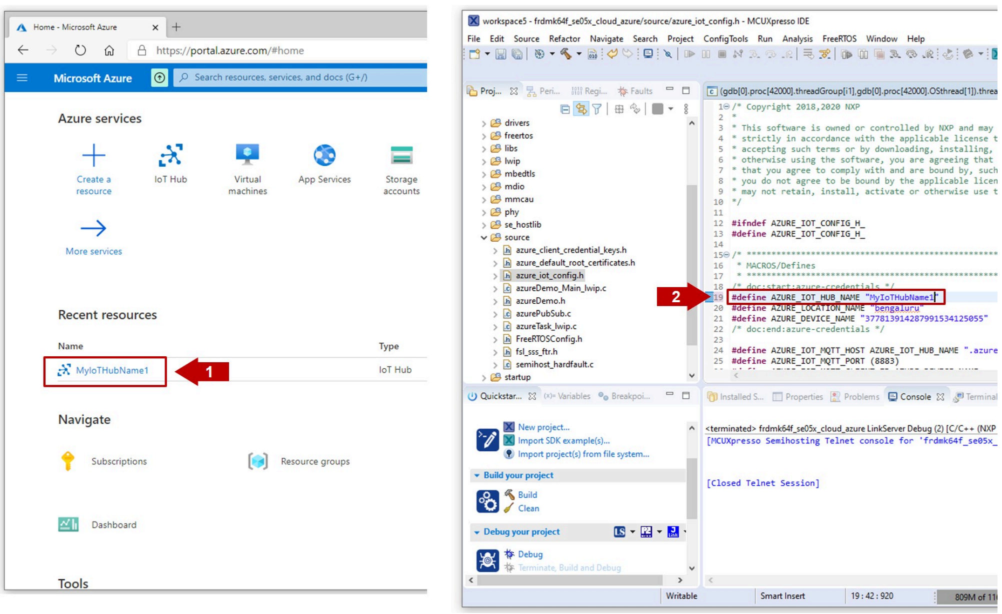


```
/* Copyright 2018 NXP
*
* This software is owned or controlled by NXP and may only be used
* strictly in accordance with the applicable license terms.  By expressly
* accepting such terms or by downloading, installing, activating and/or
* otherwise using the software, you are agreeing that you have read, and
* that you agree to comply with and are bound by, such license terms.  If
* you do not agree to be bound by the applicable license terms, then you
* may not retain, install, activate or otherwise use the software.
*/

#ifndef AZURE_IOT_CONFIG_H_
#define AZURE_IOT_CONFIG_H_

/* **************************************************************************************************
 * MACROS/Defines
 * ************************************************************************************************** */
/* doc:start:azure-credentials */
#define AZURE_IOT_HUB_NAME "QMC-TestHub"
#define AZURE_LOCATION_NAME "Hamburg"
#define AZURE_DEVICE_NAME "377815972228509164001886"
/* doc:end:azure-credentials */

#define AZURE_IOT_MQTT_HOST AZURE_IOT_HUB_NAME ".azure-devices.net" ///< Customer specific MQTT HOST.
#define AZURE_IOT_MQTT_PORT (8883)                                  ///< default port for MQTT/S
#define AZURE_IOT_MQTT_CLIENT_ID AZURE_DEVICE_NAME
```

```
No consoles to display at this time.
```

Figure 58. Find `azure_iot_config.h` file

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:



Recent resources

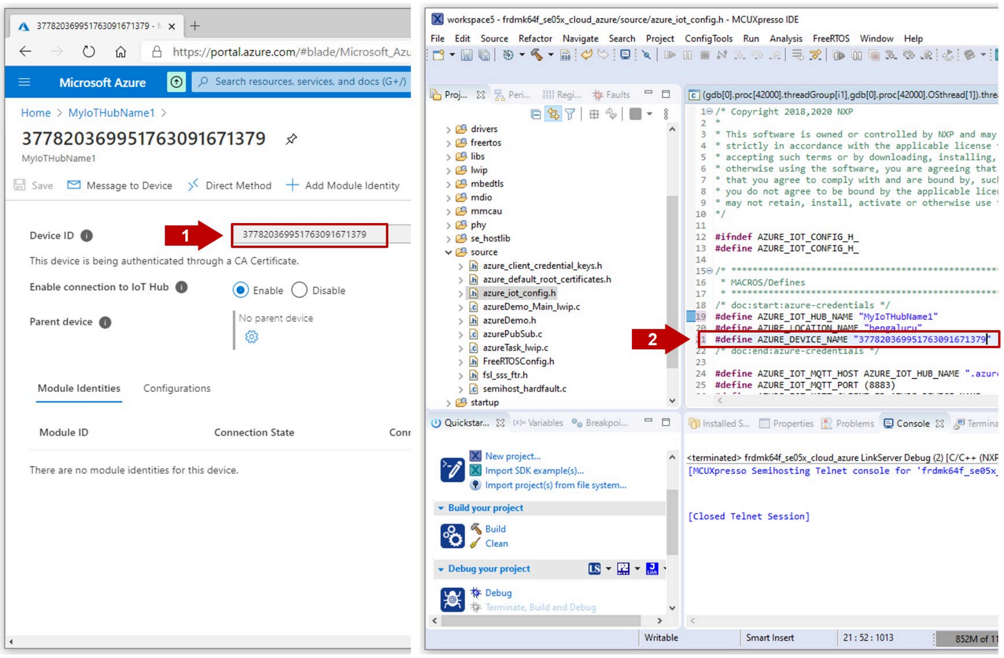
Name	Type
MyIoTHubName1	IoT Hub

```
/* Copyright 2018,2020 NXP
*
* This software is owned or controlled by NXP and may
* strictly in accordance with the applicable license t
* accepting such terms or by downloading, installing,
* otherwise using the software, you are agreeing that
* that you agree to comply with and are bound by, such
* you do not agree to be bound by the applicable licen
* may not retain, install, activate or otherwise use t
*/

#ifndef AZURE_IOT_CONFIG_H_
#define AZURE_IOT_CONFIG_H_

/* ****************************************************
 * MACROS/Defines
 * ****************************************************
/* doc:start:azure-credentials */
#define AZURE_IOT_HUB_NAME "MyIoTHubName1"
#define AZURE_LOCATION_NAME "bengaluru"
#define AZURE_DEVICE_NAME "377813914287991534125055"
/* doc:end:azure-credentials */

#define AZURE_IOT_MQTT_HOST AZURE_IOT_HUB_NAME ".azure
#define AZURE_IOT_MQTT_PORT (8883)
```

```
<terminated> frdmk64f_se05x_cloud_azure LinkServer Debug (2) [C/C++ (NXP
[MCUXpresso Semihosting Telnet console for 'frdmk64f_se05x_

[Closed Telnet Session]
```

Figure 59. Change `#define AZURE_IOT_HUB_NAME`

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:



37782036995176309167 1379

Home > MyIoTHubName1 >

37782036995176309167 1379

MyIoTHubName1

Device ID: 37782036995176309167 1379

This device is being authenticated through a CA Certificate.

Enable connection to IoT Hub: [✓ Enable] Disable

Parent device: No parent device

Module ID	Connection State

There are no module identities for this device.

workspace5 - frdmk64f_se05x_cloud_azure/source/azure_iot_config.h - MCUXpresso IDE

```
/* Copyright 2018,2020 NXP
 *
 * This software is owned or controlled by NXP and may
 * strictly in accordance with the applicable license
 * accepting such terms or by downloading, installing,
 * otherwise using the software, you are agreeing that
 * that you agree to comply with and are bound by, suc
 * you do not agree to be bound by the applicable lice
 * may not retain, install, activate or otherwise use
 */

#ifndef AZURE_IOT_CONFIG_H_
#define AZURE_IOT_CONFIG_H_

/* ****************************************************
 * MACROS/Defines
 * ****************************************************
/* doc:start:azure-credentials */
#define AZURE_IOT_HUB_NAME "MyIoTHubName1"
#define AZURE_LOCATION_NAME "bengaluru"
#define AZURE_DEVICE_NAME "37782036995176309167 1379"
/* doc:end:azure-credentials */

#define AZURE_IOT_MQTT_HOST AZURE_IOT_HUB_NAME ".azur
#define AZURE_IOT_MQTT_PORT (8883)
```

```
<terminated> frdmk64f_se05x_cloud_azure LinkServer Debug (2) [C/C++ (NXP
[MCUXpresso Semihosting Telnet console for 'frdmk64f_se05x_

[Closed Telnet Session]
```

Figure 60. Change `#define AZURE_DEVICE_NAME`

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:

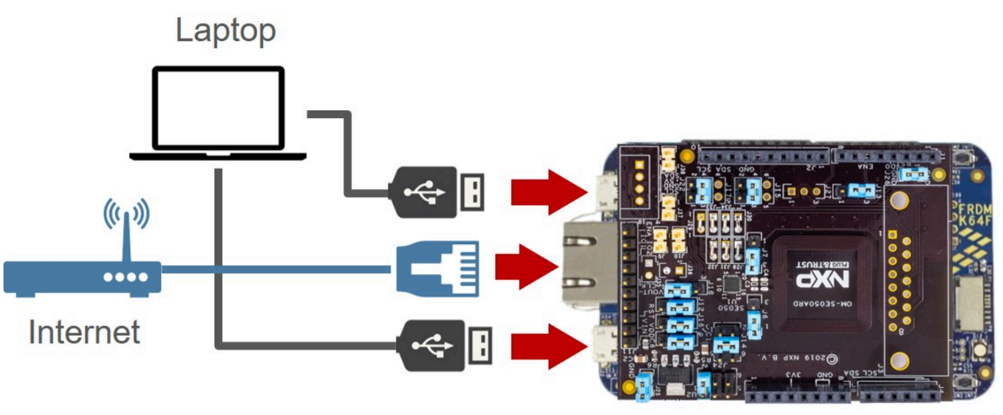
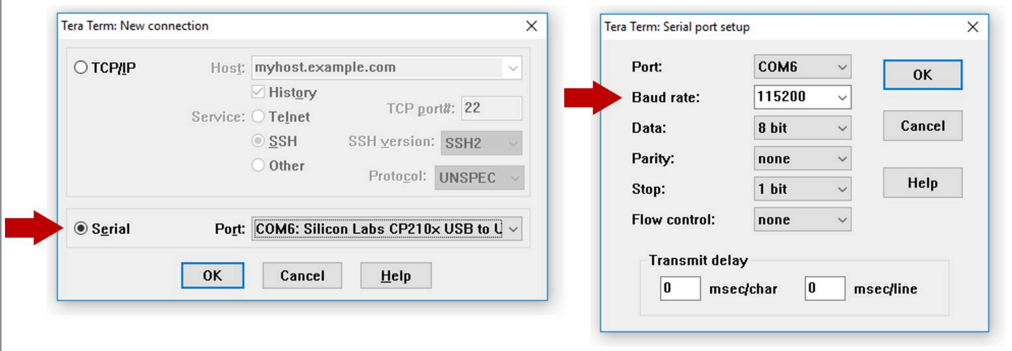


Figure 61. Connect FRDM-K64F board

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.



Tera Term: New connection

○ TCP/IP	Host:	myhost.example.com
	☑ History	
Service:	○ Telnet	TCP port#: 22
	● SSH	SSH version: SSH2
	○ Other	Protocol: UNSPEC
● Serial	Port:	COM6: Silicon Labs CP210x USB to U

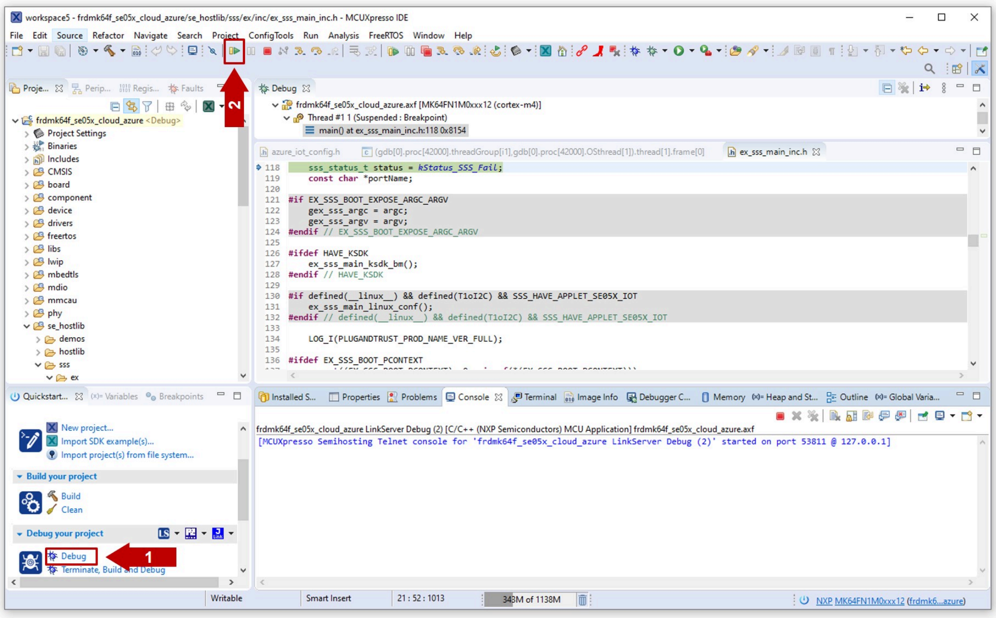
Tera Term: Serial port setup

Port:	COM6
Baud rate:	115200
Data:	8 bit
Parity:	none
Stop:	1 bit
Flow control:	none

Transmit delay: 0 msec/char 0 msec/line

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:



workspace5 - frdmk64f_se05x_cloud_azure/se_hostlib/sss/ex/inc/ex_sss_main_inc.h - MCUXpresso IDE

File Edit Source Refactor Navigate Search Project ConfigTools Run Analysis FreeRTOS Window Help
2
1
Proje... Perip... Regis... Faults
frdmk64f_se05x_cloud_azure <Debug>
Project Settings
Binaries
Includes
CMSIS
board
component
device
drivers
freertos
libs
lwip
mbedtls
mdio
mmcau
phy
se_hostlib
demos
hostlib
sss
ex
Quickstart... Variables Breakpoints
New project...
Import SDK example(s)...
Import project(s) from file system...
Build your project
Build
Clean
Debug your project
Debug
Terminate, Build and Debug
Debug
Installed S... Properties Problems Console Terminal Image Info Debugger C... Memory Heap and St... Outline Global Varia...
Writable
Smart Insert
21 : 52 : 1013
343M of 1138M
NXP MK64FN1M0xxx12 (frdmk6...azure)

```
frdmk64f_se05x_cloud_azure.axf [MK64FN1M0xxx12 (cortex-m4)]
  Thread #1 1 (Suspended : Breakpoint)
    main() at ex_sss_main_inc.h:118 0x8154
```

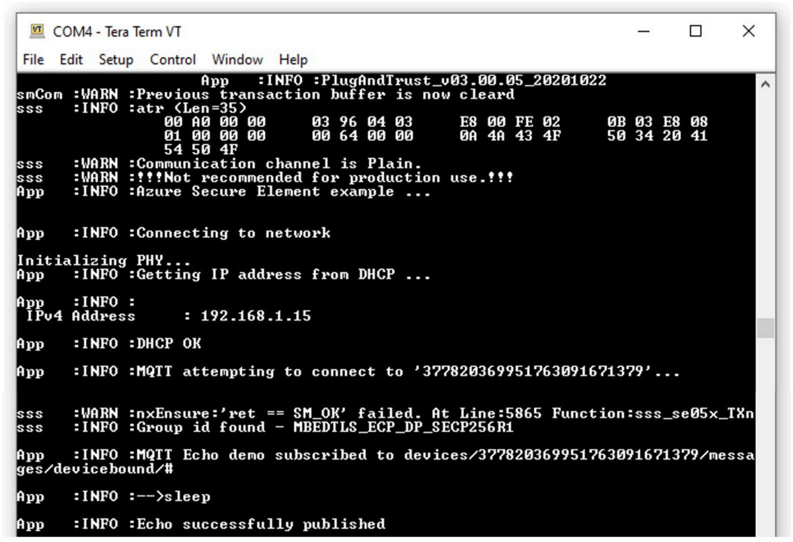
azure_iot_config.h | (gdb[0].proc[42000].threadGroup[i1],gdb[0].proc[42000].OSthread[1]).thread[1].frame[0] | ex_sss_main_inc.h

```
    sss_status_t status = kStatus_SSS_Fail;
    const char *portName;

#if EX_SSS_BOOT_EXPOSE_ARGC_ARGV
    gex_sss_argc = argc;
    gex_sss_argv = argv;
#endif // EX_SSS_BOOT_EXPOSE_ARGC_ARGV

#ifdef HAVE_KSDK
    ex_sss_main_ksdk_bm();
#endif // HAVE_KSDK

#if defined(__linux__) && defined(T1oI2C) && SSS_HAVE_APPLET_SE05X_IOT
    ex_sss_main_linux_conf();
#endif // defined(__linux__) && defined(T1oI2C) && SSS_HAVE_APPLET_SE05X_IOT

    LOG_I(PLUGANDTRUST_PROD_NAME_VER_FULL);

#ifdef EX_SSS_BOOT_PCONTEXT
```

```
frdmk64f_se05x_cloud_azure LinkServer Debug (2) [C/C++ (NXP Semiconductors) MCU Application] frdmk64f_se05x_cloud_azure.axf
[MCUXpresso Semihosting Telnet console for 'frdmk64f_se05x_cloud_azure LinkServer Debug (2)' started on port 53811 @ 127.0.0.1]
```

Figure 63. Debug Azure IoT Hub project

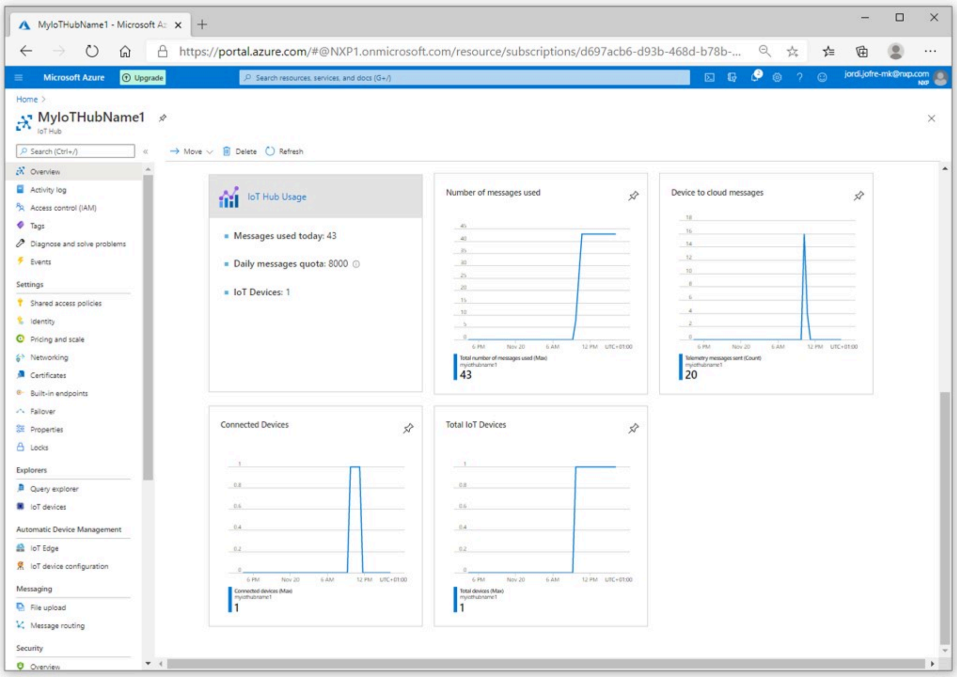
4. Your device should now be connected to Azure IoT Hub. Check that your device is connected by:
   a. Checking the TeraTerm logs as shown in Figure 64.



```
                    App    :INFO :PlugAndTrust_v03.00.05_20201022
smCom  :WARN :Previous transaction buffer is now cleard
sss    :INFO :atr (Len=35)
                 00 A0 00 00     03 96 04 03     E8 00 FE 02     0B 03 E8 08
                 01 00 00 00     00 64 00 00     0A 4A 43 4F     50 34 20 41
                 54 50 4F
sss    :WARN :Communication channel is Plain.
sss    :WARN :!!!Not recommended for production use.!!!
App    :INFO :Azure Secure Element example ...

App    :INFO :Connecting to network

Initializing PHY...
App    :INFO :Getting IP address from DHCP ...

App    :INFO :
 IPv4 Address      : 192.168.1.15

App    :INFO :DHCP OK

App    :INFO :MQTT attempting to connect to '377820369951763091671379'...


sss    :WARN :nxEnsure:'ret == SM_OK' failed. At Line:5865 Function:sss_se05x_TXn
sss    :INFO :Group id found - MBEDTLS_ECP_DP_SECP256R1

App    :INFO :MQTT Echo demo subscribed to devices/377820369951763091671379/messa
ges/devicebound/#

App    :INFO :-->sleep

App    :INFO :Echo successfully published
```

Figure 64. Device connection to Azure IoT Hub

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



IoT Hub Usage
- Messages used today: 43
- Daily messages quota: 8000
- IoT Devices: 1

Number of messages used — Total number of messages used (Max) myiothubname1: 43

Device to cloud messages — Telemetry messages sent (Count) myiothubname1: 20

Connected Devices — Connected devices (Max) myiothubname1: 1

Total IoT Devices — Total devices (Max) myiothubname1: 1

Figure 65. Device connection to Azure IoT Hub - dashboard

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 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 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 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 the use of 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 — 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 throughout their lifecycles to reduce the effect 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

Tab. 1. OM-SE050ARD development kit details 5
Tab. 2. i.MX6Ultralite 5
Tab. 3. ECC256 public key used for Azure IoT Hub device onboarding 6
Tab. 4. OM-SE050ARD development kit details 31
Tab. 5. FRDM-K64F details 31

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 ssscli commands to retrieve credentials. 8
Fig. 5. Execution of ssscli 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 completition 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 completition 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_imx_register script. 25
Fig. 28. Device registered in Azure platform. 26
Fig. 29. CloudConn .txt file creation. 27
Fig. 30. CloudCon .txt file contents 27
Fig. 31. Output of the azure_imx_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 certificated 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

Date of release: 7 December 2020
Document identifier: AN12402
Document number: 535020