AN12404

EdgeLock™ SE05x for secure connection to AWS IoT Core

Rev. 1.6 — 15 January 2021 Application note

535216

Document information

Information	Content
Keywords	EdgeLock SE05x, AWS IoT Core, Secure cloud onboarding
Abstract	This application note describes how to leverage the EdgeLock SE05x for secure cloud onboarding to the AWS IoT Core 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 FRDM-K64F board

Revision history

Revision history

Revision number	Date	Description
1.0	2019-07-17	First release
1.1	2019-11-26	Update referring to MW v02.12 KSDK
1.2	2020-02-27	Updated to MW v02.12.03 and added appendix for Ease of Use configuration
1.3	2020-04-30	Fixed AWS thing subscription topic
1.4	2020-11-11	Updated document to include AWS IoT Core Multi-Account Registration feature
1.5	2020-12-07	Updated to latest template and fixed broken URLs
1.6	2021-01-15	Updated Section 4.1.3

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 AWS IoT Core can be trusted by the OEM. AWS IoT Core 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 AWS IoT Core. Using the EdgeLock SE05x security IC, OEMs can safely connect their devices to AWS IoT Core 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 AWS IoT Core.

Note: *NXP is offering the EdgeLock 2GO service, which supports different options for provisioning your devices and onboarding your devices to AWS IoT Core. You can find more details about EdgeLock 2GO at* www.nxp.com/edgelock2go.

2 Leveraging EdgeLock SE05x for AWS IoT Core device onboarding

The security architecture of the AWS IoT Core uses X.509 certificates and TLS authentication for device onboarding. AWS IoT Core implements a feature called *Multi-Account Registration*, which simplifies the device registration process and makes it possible to onboard devices without requiring the registration of a CA certificate in AWS IoT Core.

The EdgeLock SE05x is pre-qualified to work with AWS IoT Core *Multi-Account Registration* feature, meaning that the pre-provisioned credentials in EdgeLock SE05x are qualified to connect to AWS IoT Core by default,. This way, devices can connect to AWS IoT Core by just registering the device certificate stored in EdgeLock SE05x.

Figure 1 illustrates the device registration flow using EdgeLock SE05x ease of use configuration:

1. NXP delivers a quantity of EdgeLock SE05x ICs based on a purchase order to the OEM's manufacturing 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 AWS IoT Core account and registers on it every device by registering its device certificate.
4. IoT devices boot up and automatically connect to AWS IoT Core service using the pre-provisioned credentials inside EdgeLock SE05x.

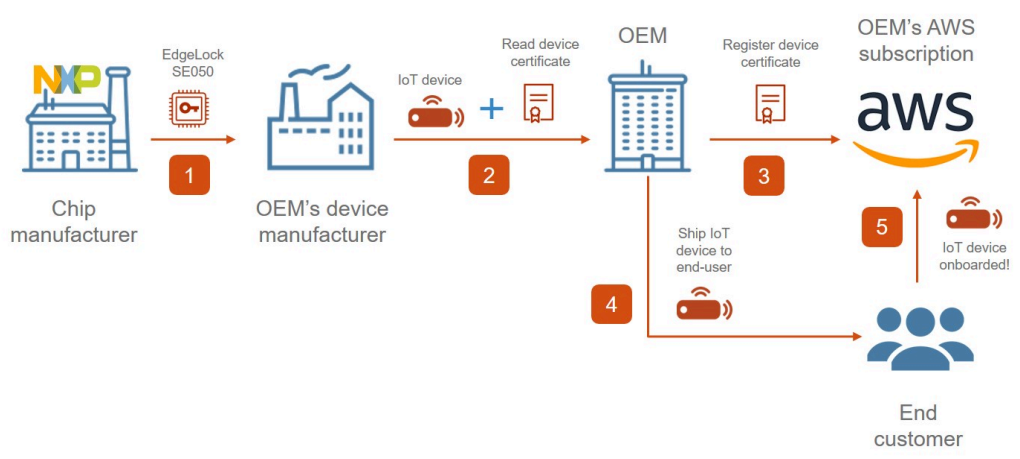

Figure 1. AWS IoT Core device registration flow

3 Running AWS IoT Core device onboarding project example

The AWS IoT Core project example showcases how to leverage EdgeLock SE05x security IC to set up trusted connections to AWS IoT Core cloud. This section explains how to run the AWS project example included as part of the EdgeLock SE05x support package.

Note: *The AWS device onboarding procedure described in this section and the AWS 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 AWS IoT Core project example using the hardware described below. However, you could use other MCU / MPU boards supported by EdgeLock SE05x Plug & Trust Middleware for this purpose as well.

1. OM-SE050ARD development kit:

Table 1. OM-SE050ARD development kit details

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

2. FRDM-K64F board:

Table 2. FRDM-K64F details

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

3.2 Sign up for an AWS IoT Core account

Amazon offers 12 months of free tier access. To create an AWS IoT Core account:

1. Go to https://aws.amazon.com/iot-core/ and click *Get started for free* button as shown in Figure 2:

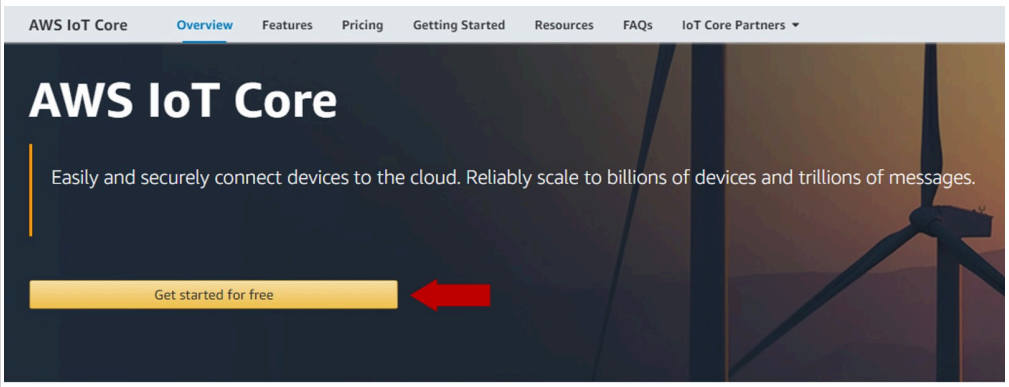

Figure 2. Get started with AWS IoT Core for free

2. If you already have an account with AWS, you will be prompted to log in. If you do not have an account yet, click on *Create a new AWS account* as shown in Figure 3:

Figure 3. Sign in or create a new AWS IoT Core account

3. Next, fill in the form with your email, password, AWS account name and click *Continue* as shown in Figure 4:

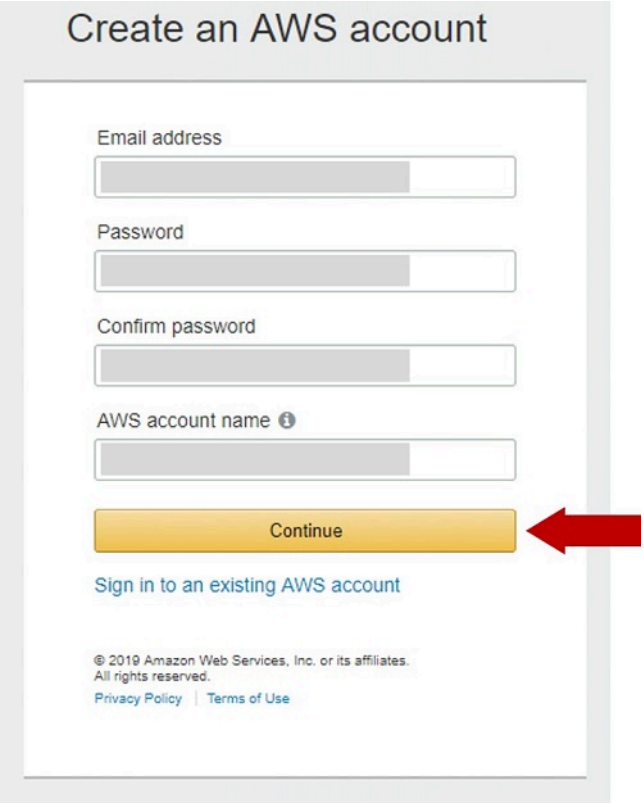

Figure 4. Create an AWS IoT Core account

4. Select the account type, complete the fields with your contact details and click *Create Account and Continue* as shown in Figure 5:

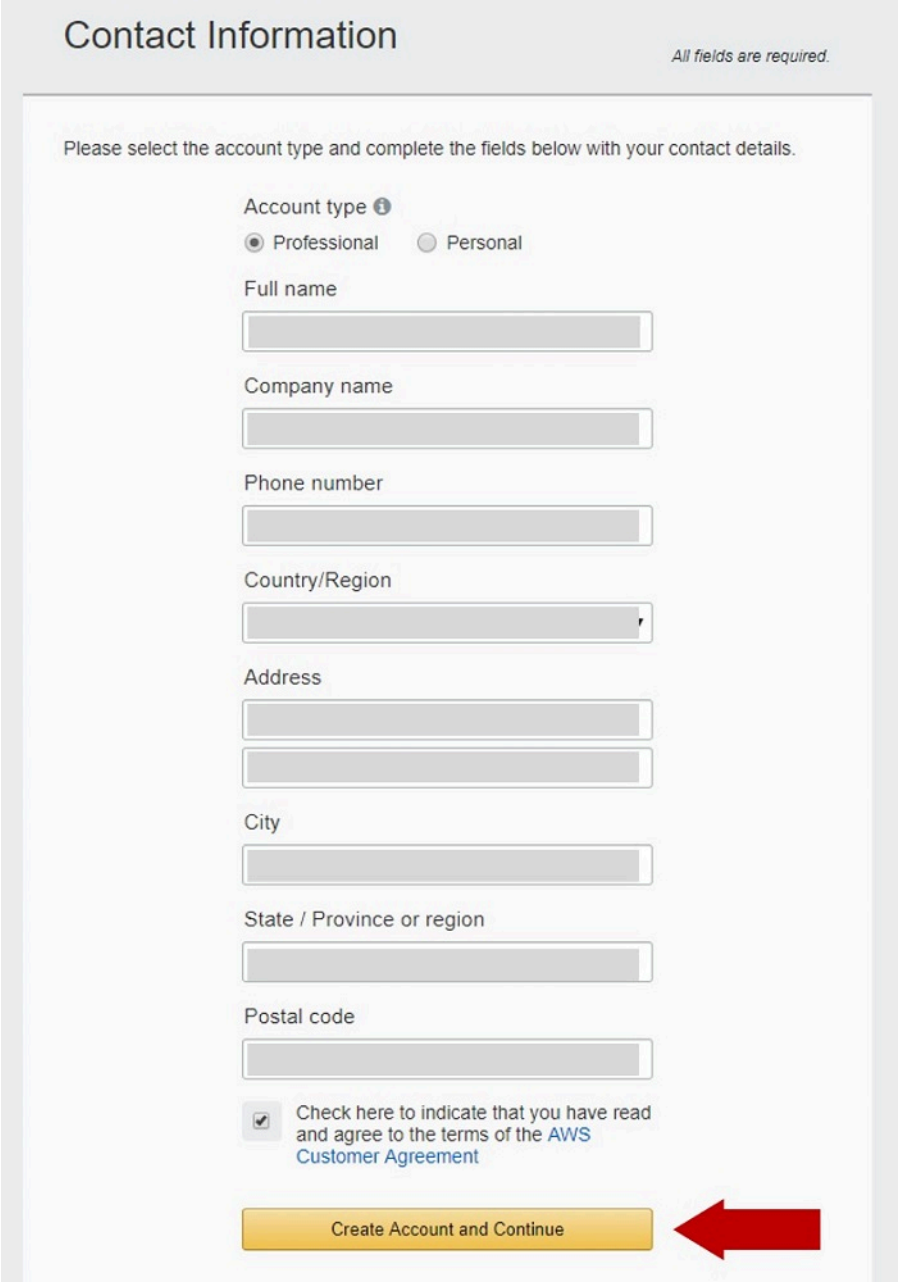

Figure 5. Create an AWS IoT Core account - Contact information

5. Supply a valid credit or debit card and click ***Secure Submit*** as shown in Figure 6. AWS will use it to verify your identity (i.e.there might be a record for a $1 transaction on your bank statement that will be automatically returned). AWS will not charge you

unless your usage exceeds the AWS Free Tier Limits. You can check the limits in Free Tier Limits.

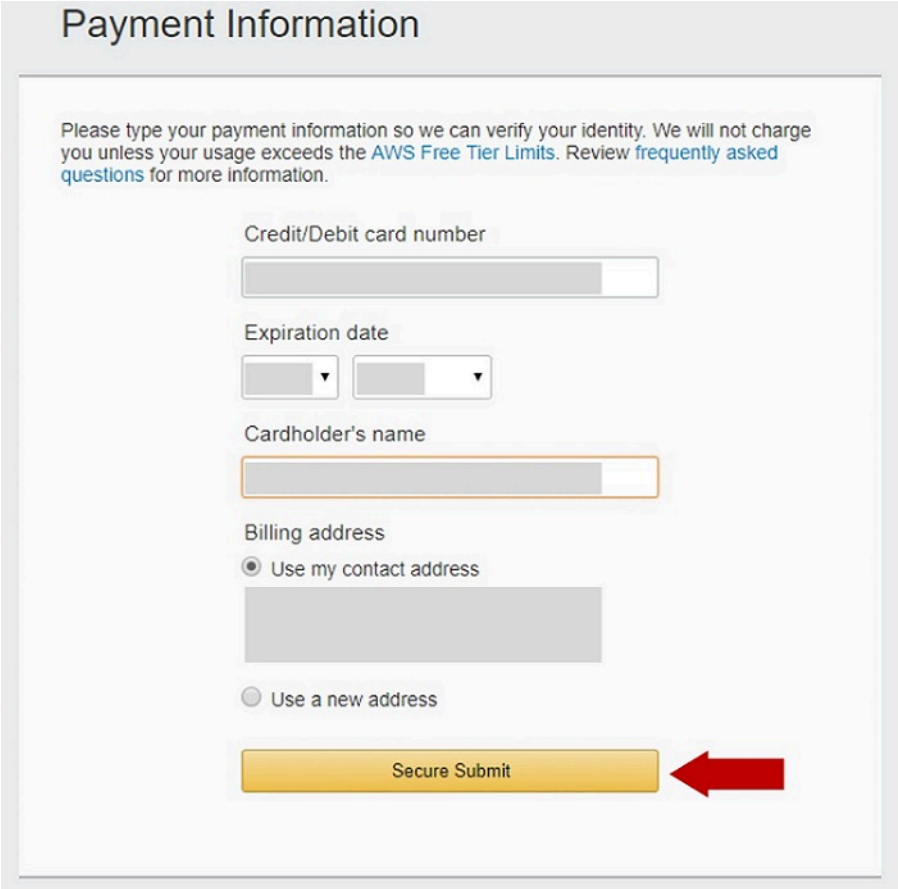

Figure 6. Create an AWS IoT Core account - Payment information

6. Verify your phone number to confirm your identity as shown in Figure 7. When you continue, the AWS automated system will contact you with a verification code.

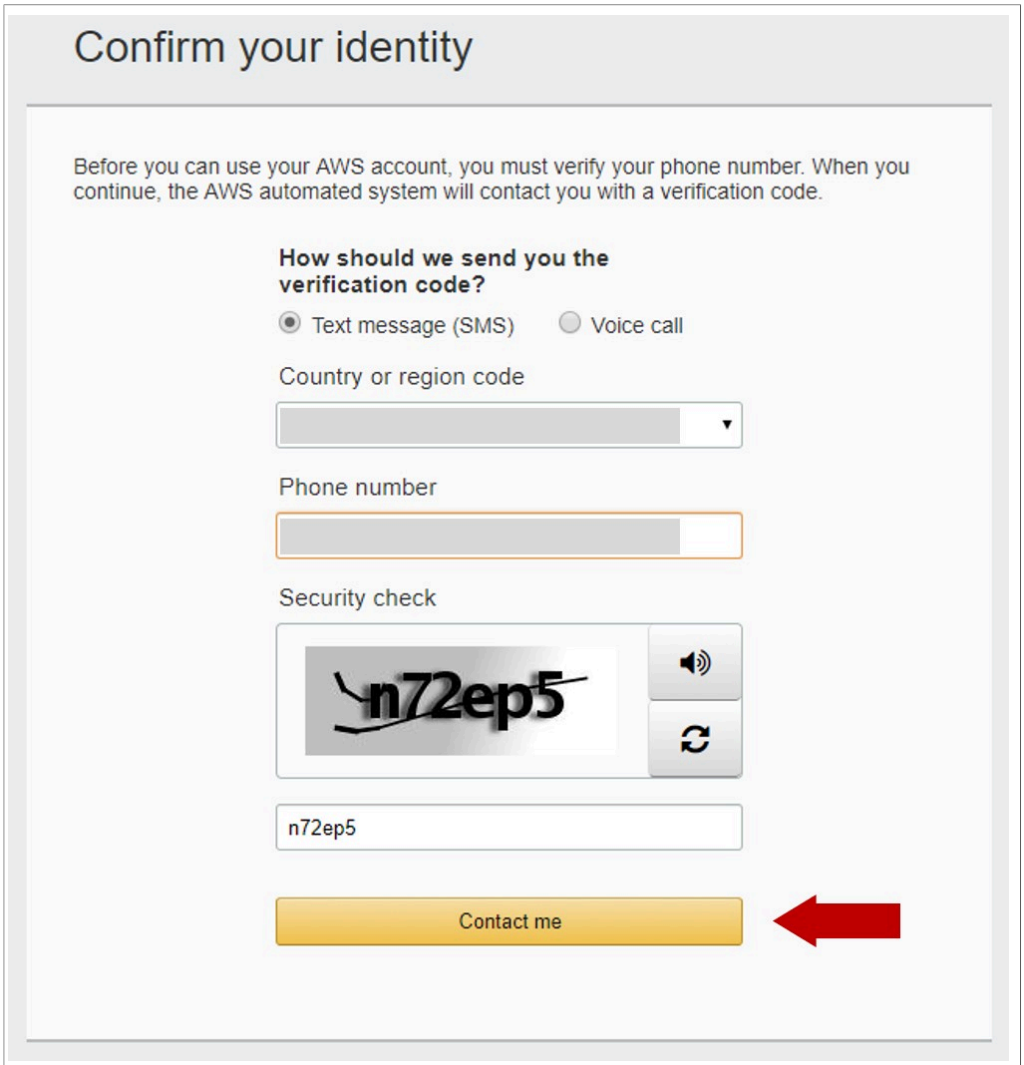

Figure 7. Create an AWS IoT Core account - Confirm your identity

7. Enter the verification code that was sent to your cell phone and click *Verify code* as shown in Figure 8:

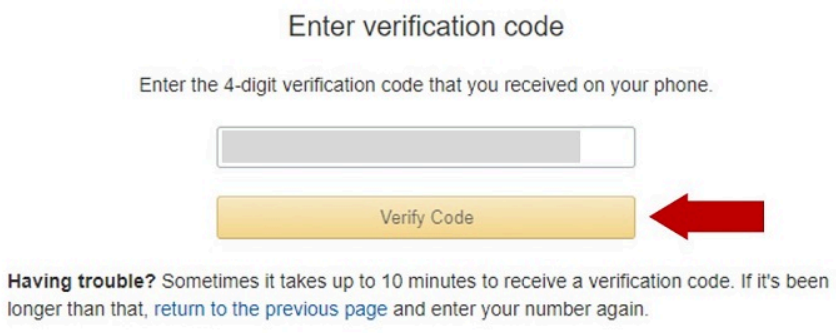

Figure 8. Create an AWS IoT Core account - Enter verification code

8. Your identity is validated. Choose *Basic* to obtain a free subscription as show in Figure 9

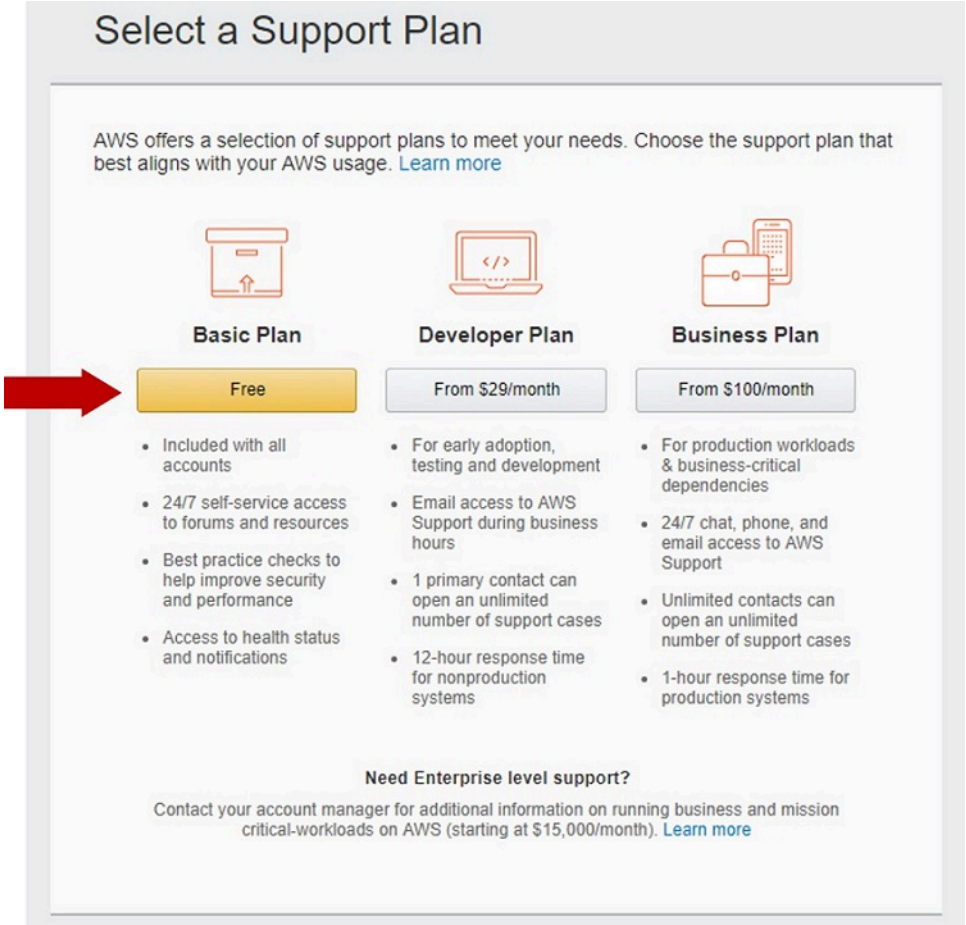

Figure 9. Create an AWS IoT Core account - Select a support plan

9. In a few minutes, you will receive via email the account creation confirmation. On the welcome page, choose *Get started* as shown in Figure 10:

Figure 10. AWS Management Console - IoT landing page

10. To get started with AWS IoT Core, click *Services* and then *IoT Core* from the AWS Management Console as shown in Figure 11:

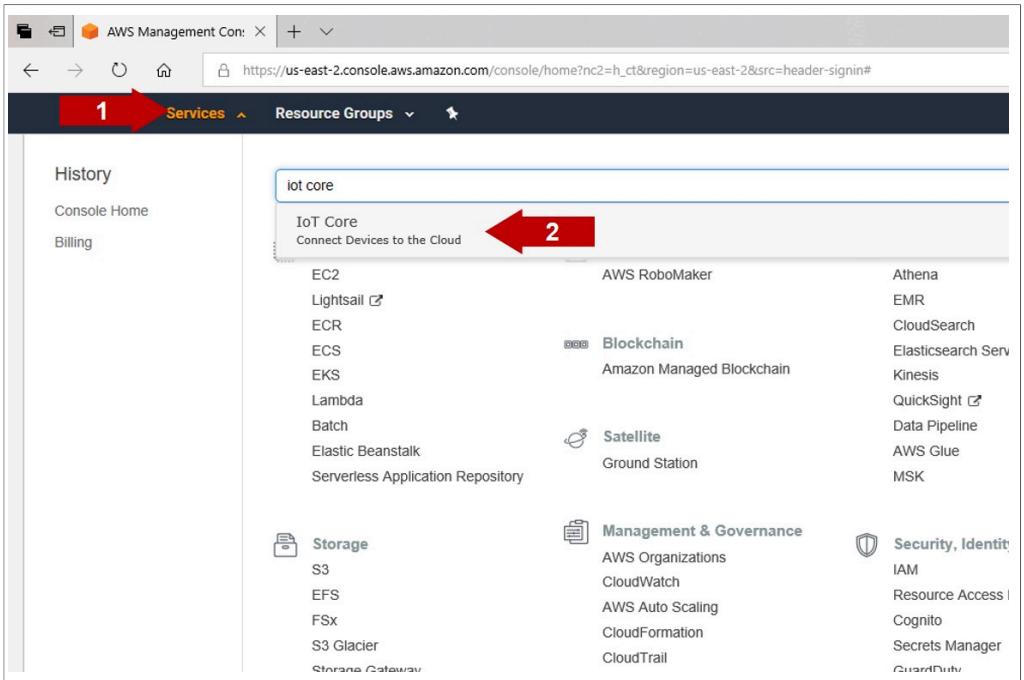

Figure 11. AWS Management Console - AWS IoT Core

3.3 Create an AWS IoT thing

An AWS IoT thing is a representation of your physical device in the cloud. The AWS IoT thing is an entry in the registry that contains attributes that describe a device. Any

physical device needs a thing record in order to work with AWS IoT. To create an AWS IoT thing, follow these steps:

1. From the AWS IoT Core dashboard, go to **Manage**, go to **Things** and click on the **Register a thing** button as shown in Figure 12:

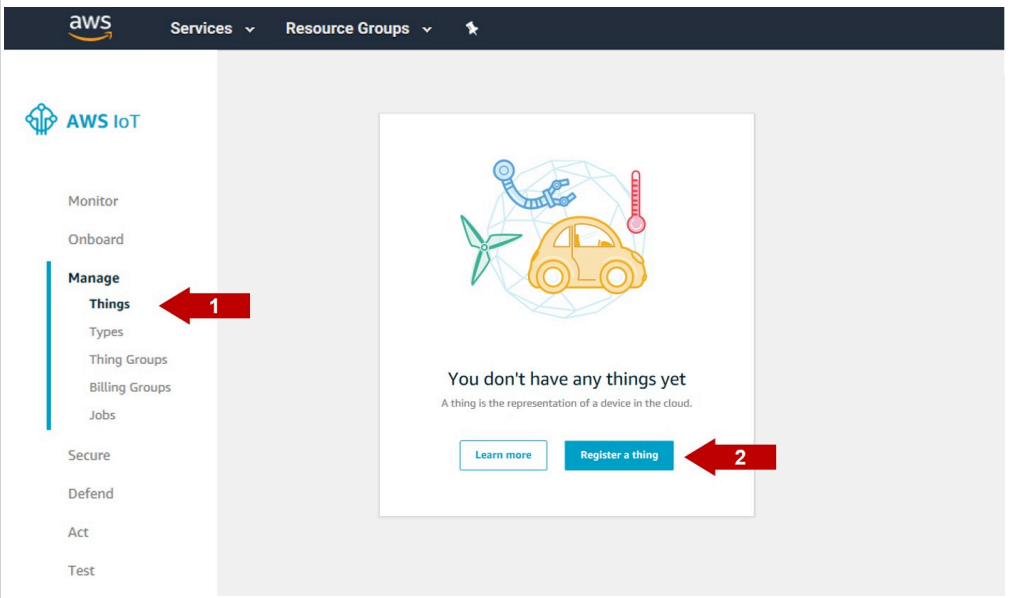

Figure 12. Go to the AWS IoT Things menu

2. A new menu called *Creating AWS IoT Things* will be opened. Click on the **Create a single thing** option as shown in Figure 13

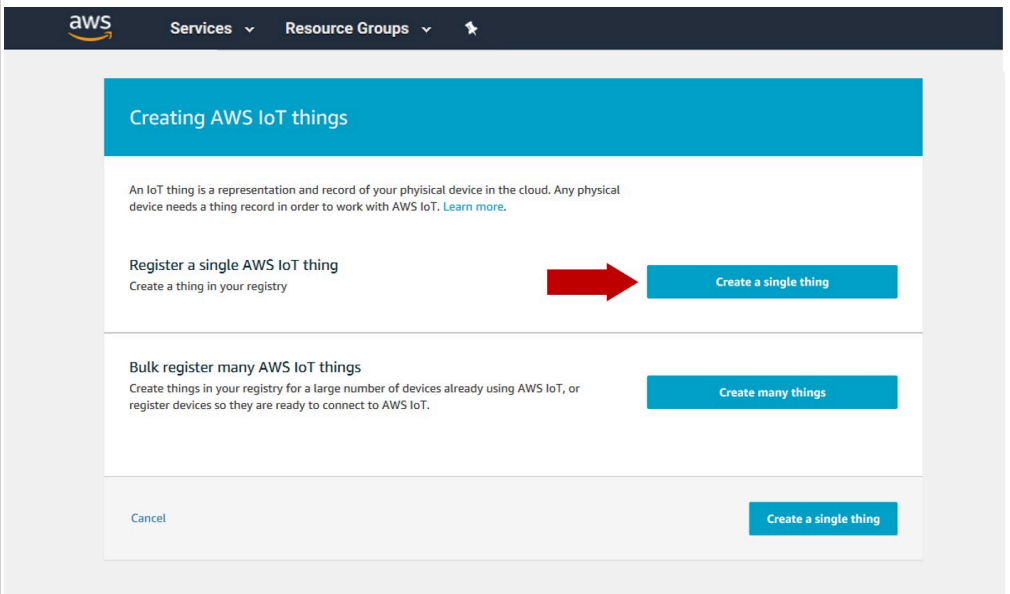

Figure 13. Select *Create a single thing* option

3. A new form called **Add your device to the thing registry** will be opened. For the purpose of this demo, you only need to fill in the AWS IoT Thing **name** and click **Next** as shown in Figure 14

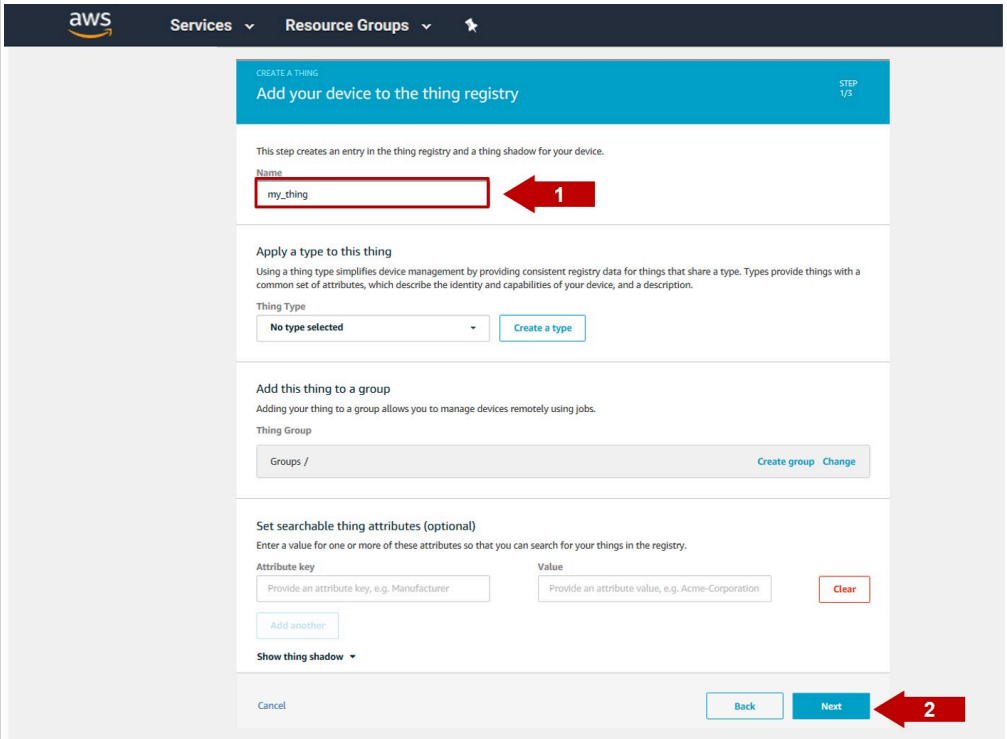

Figure 14. Add your device to the thing registry

4. Click on **Create a thing without certificate** to complete the AWS IoT Thing creation as shown in Figure 15. The certificate for your IoT Thing will be added later on in this tutorial.

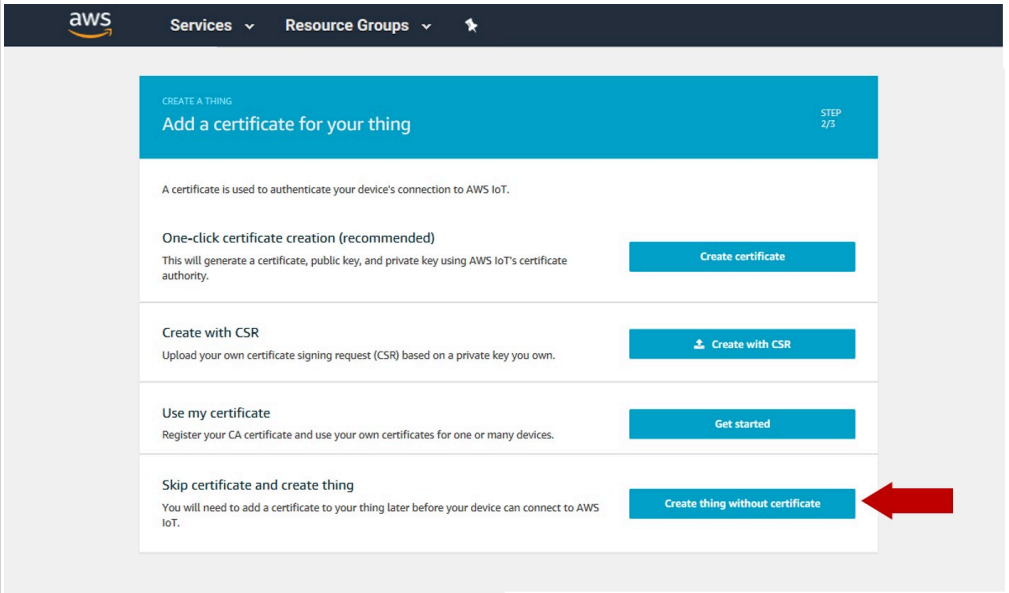

Figure 15. Create a thing without certificate

5. Now, your AWS IoT Thing should be created and visible in your AWS IoT Core dashboard as shown in Figure 16

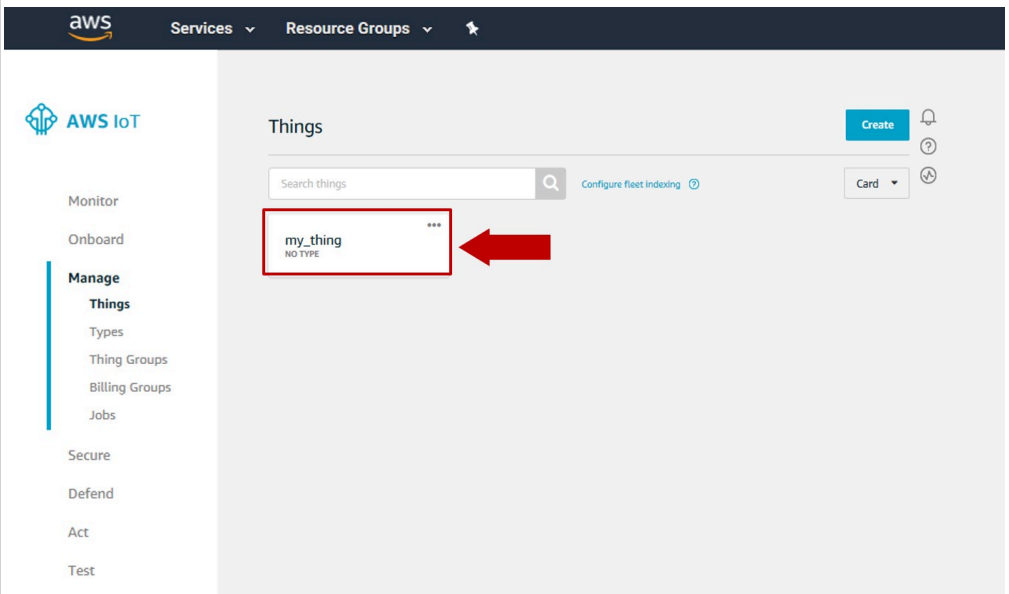

Figure 16. Confirm AWS IoT Thing creation

3.4 Create a policy

AWS IoT policies are used to authorize your device to perform AWS IoT operations, such as subscribing or publishing to MQTT topics. To allow your device to perform AWS IoT

operations, you must create an AWS IoT policy and attach it to your device certificate. To create an AWS IoT policy, follow these steps:

1. From the AWS IoT Core dashboard, go to **Secure**, go to **Policies** and click on the **Create a policy** button as shown in Figure 17:

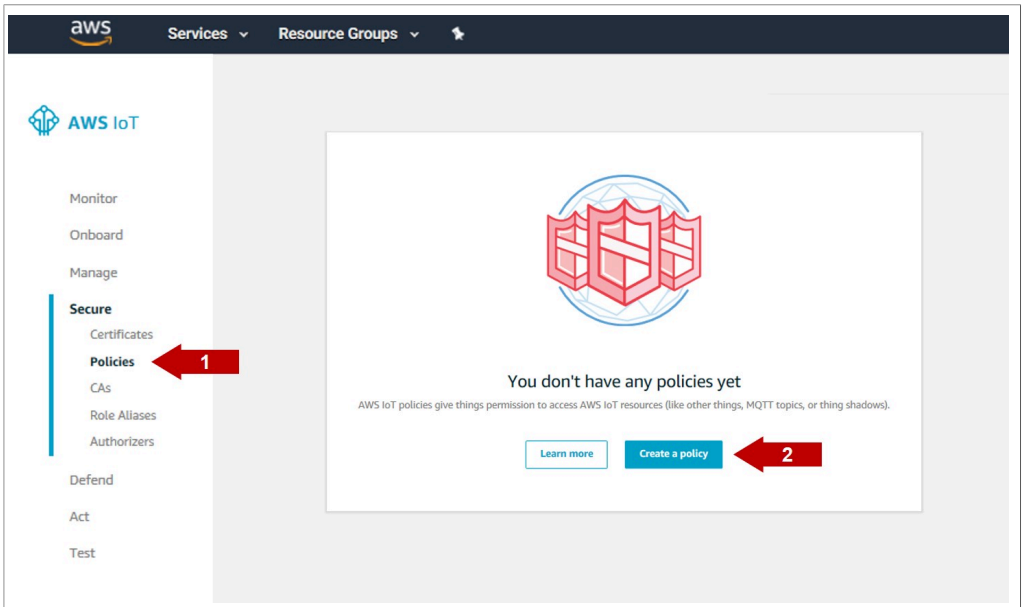

Figure 17. Go to the AWS IoT policies menu

2. A new menu called *Creating a policy* will be opened. Fill in a **name** for your AWS IoT policy and click on **Advanced mode** option as shown in Figure 18

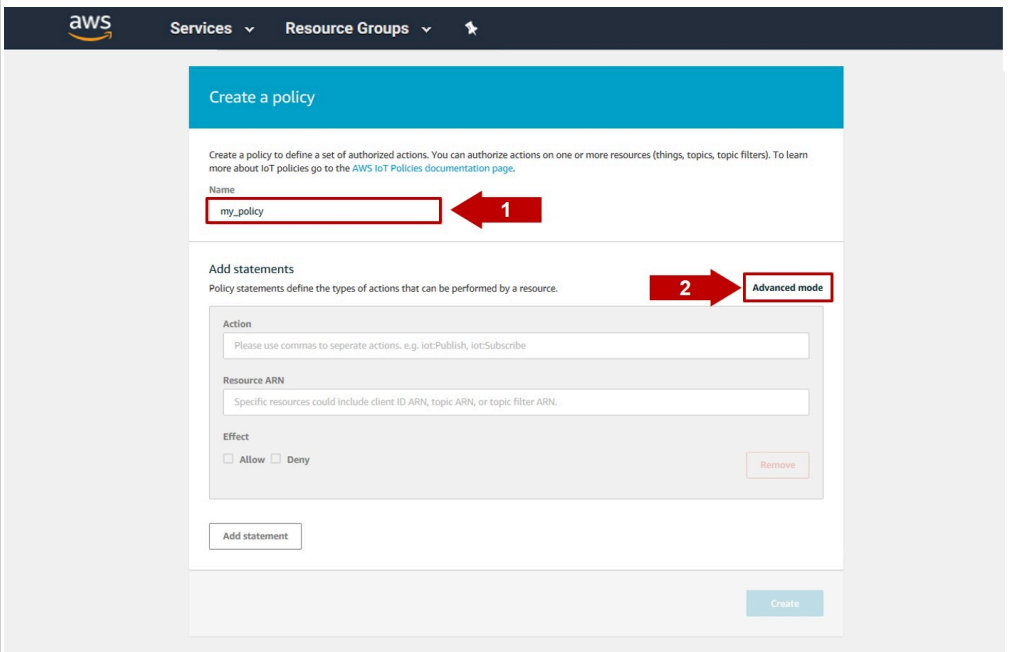

Figure 18. Create a policy name and go to Advanced mode.

3. Use the text box to personalize your policy with the following text:

```
{
"Version": "2012-10-17",
"Statement": [
{
"Effect": "Allow",
"Action": "iot:*",
"Resource": "*"
}]
}
```

Click on **Create** button as shown in Figure 19.

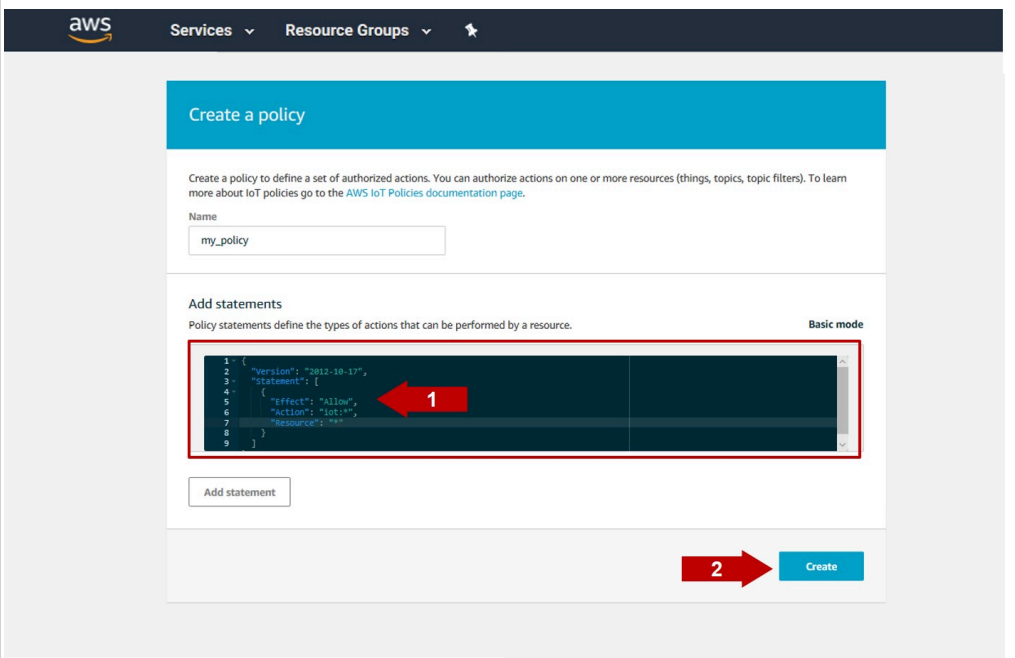

Figure 19. Personalize your AWS IoT Core policy

4. Now, your AWS IoT policy should be created and visible in your AWS IoT Core dashboard as shown in Figure 20

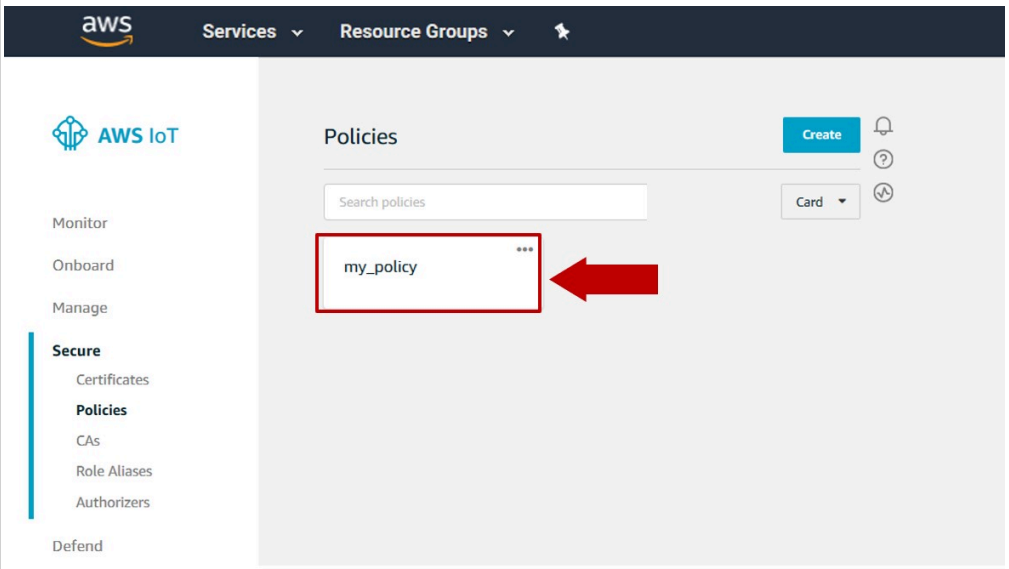

Figure 20. Confirm AWS IoT policy creation

3.5 Extracting credentials from EdgeLock SE05x

We will be using ECC credentials in this example, corresponding to key ID `0xF0000000` and certificate ID `0xF0000001`. You can use any of the available certificates that are pre-provisioned in your EdgeLock SE05x. Please refer to AN12436 - SE050 Configurations for a list of available key and certificate IDs.

3.5.1 Download EdgeLock SE05x Plug & Trust Middleware

Follow these steps to download the EdgeLock SE05x Plug & Trust Middleware in your local machine:

1. Download EdgeLock SE05x Plug & Trust Middleware from the NXP website.

2. Create a folder called ***se050_middleware*** in C: directory as shown in Figure 21:

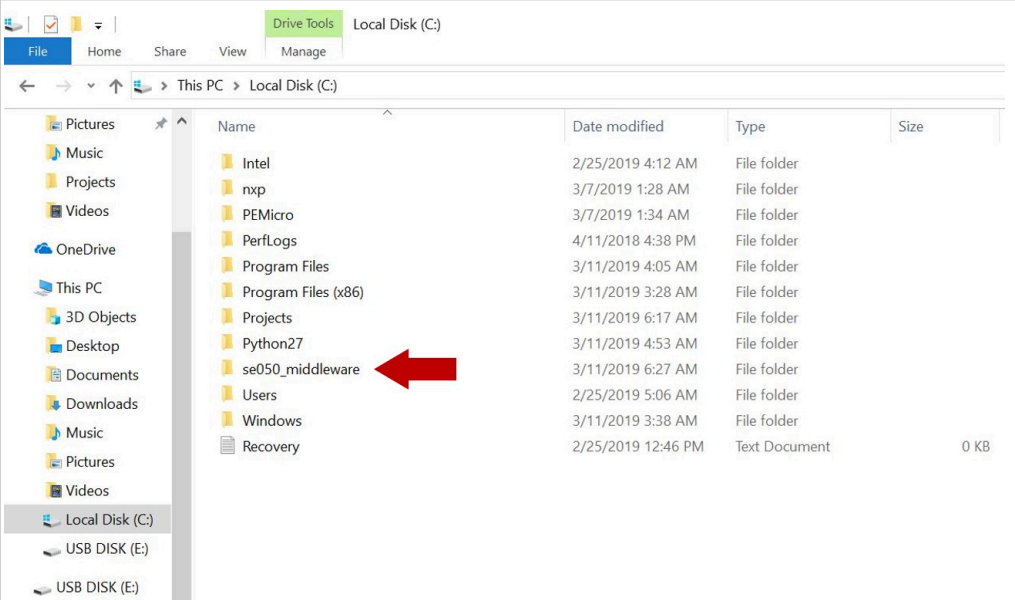

Figure 21. Create se050_middleware folder

3. Unzip the EdgeLock SE05x Plug & Trust Middleware inside the *se050_middleware* folder. After unzipping, you will see a folder called ***simw-top*** created. The contents of the ***simw-top*** directory should look as shown in Figure 22:

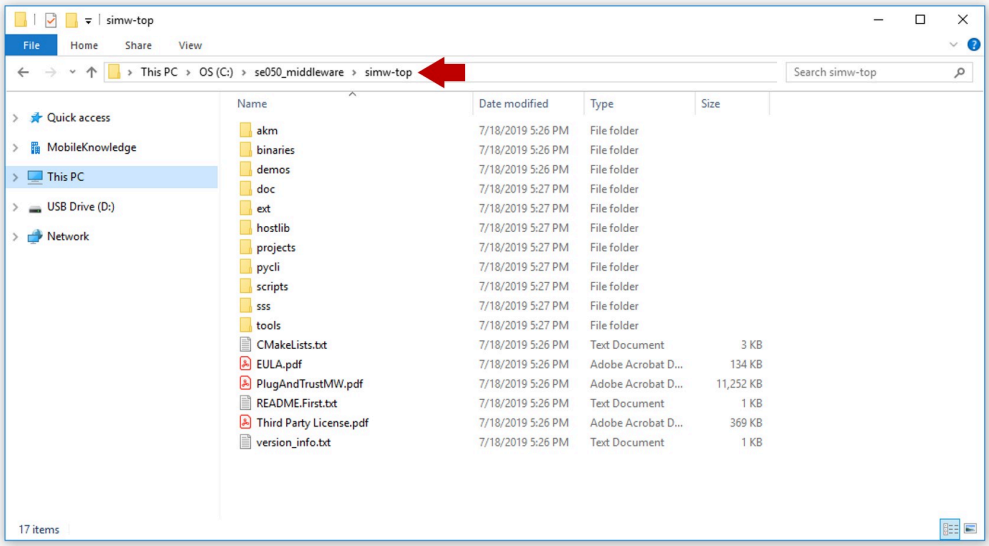

Figure 22. Unzip se050 middleware

Note: *It is recommended to keep* `se050_middleware` *with the* ***shortest*** *path possible and* ***without spaces*** *in it. This avoids some issues that could appear when building the middleware if the path contains spaces.*

3.5.2 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 23:

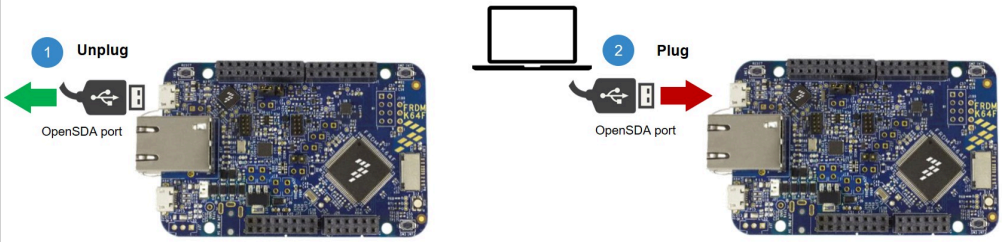

Figure 23. Unplug and plug OpenSDA port

2. When you plug the board, your laptop should recognize the board as an external drive as shown in Figure 24:

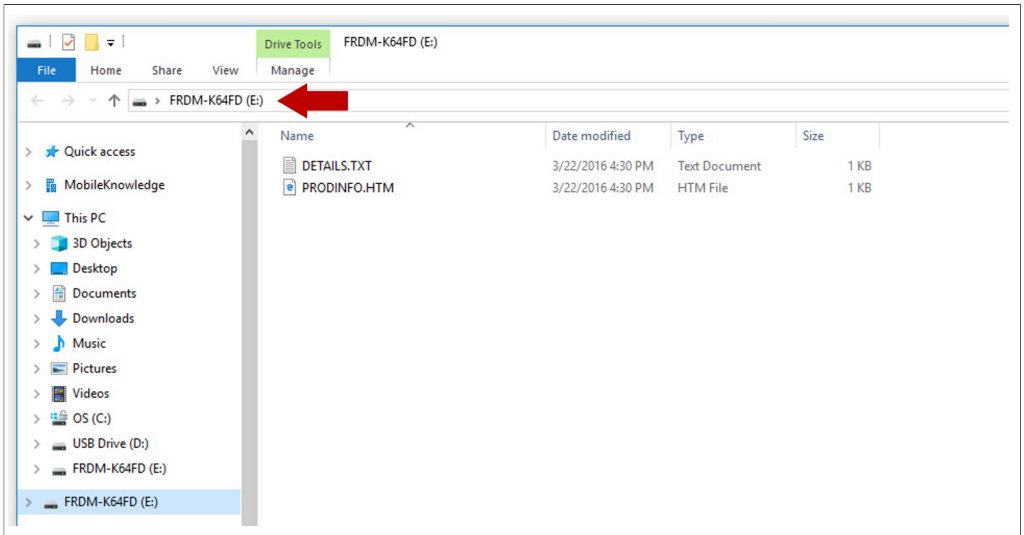

Figure 24. FRDM-K64F drive

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 25:

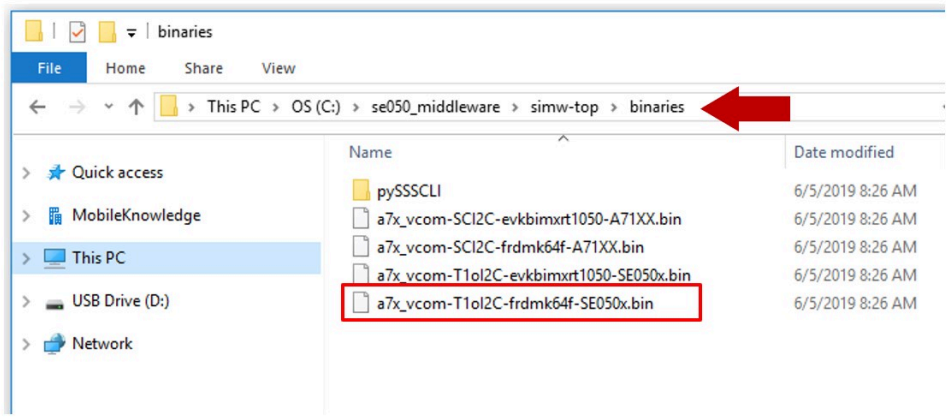

Figure 25. VCOM binary folder

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 26:

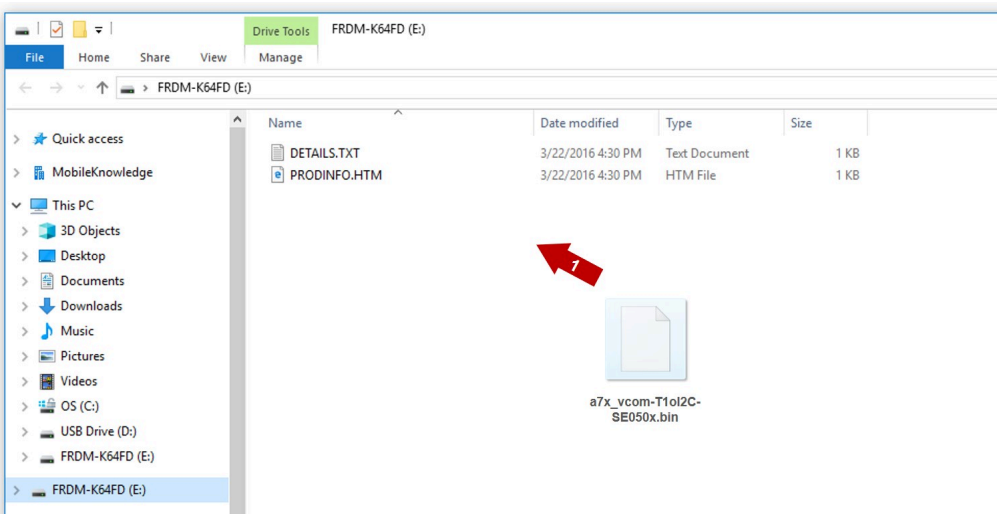

Figure 26. Drag and drop VCOM binary

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 27:
   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.

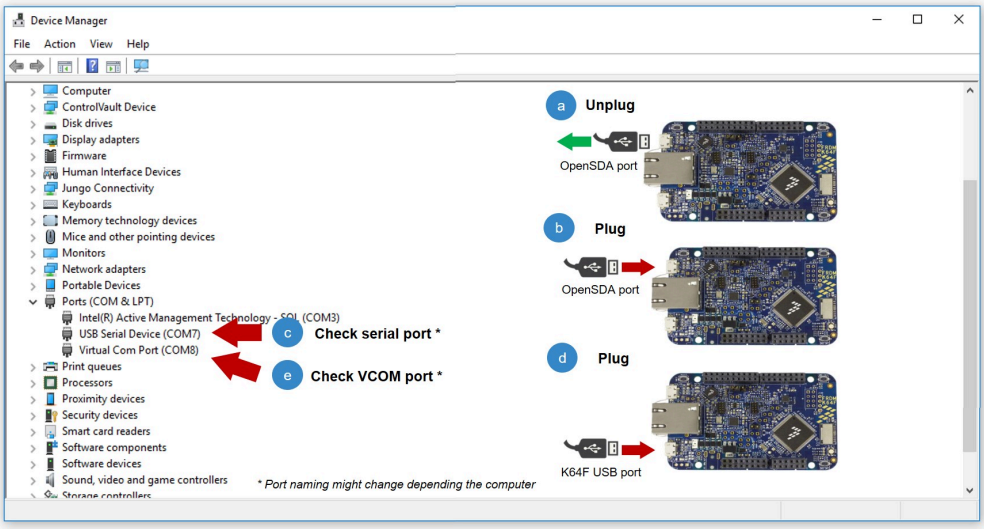

Figure 27. Check VCOM and serial ports

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.*

3.5.3 Read device certificate from EdgeLock SE05x

To read the device certificate from EdgeLock SE05x storage, follow these steps:

1. First, open a command prompt and navigate to `C:\se050_middleware\simw-top\binaries\pySSSCLI`.
   Send `>cd C:\se050_middleware\simw-top\binaries\pySSSCLI`.
2. Connect to the EdgeLock SE05x using the executable `ssscli.exe`. You need to indicate the VCOM port number corresponding to the K64 USB port of your board (See Section 3.5.2).
   Send `>ssscli connect se050 vcom COM9`.

3. We shall first make sure that the chosen keys we will be using are indeed available in the EdgeLock SE05x. To do this, we will fetch a list of available keys.
   Send `>ssscli se05x readidlist`. As shown in Figure 28, we can confirm that the desired keys are available.

```
C:\Windows\System32\cmd.exe
C:\se050_middleware\simw-top\binaries\pySSSCLI>ssscli connect se050 vcom COM9

C:\se050_middleware\simw-top\binaries\pySSSCLI>ssscli se05x readidlist
Opening COM Port '\\.\COM9'
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.!!!
Key-Id: 0X7da00013   NIST-P          (Public Key)   Size(Bits): 256
Key-Id: 0X7da00003   NIST-P          (Public Key)   Size(Bits): 256
Key-Id: 0X7da00012   AES                            Size(Bits): 128
Key-Id: 0X7da00002   AES                            Size(Bits): 128
Key-Id: 0X7da00011   USER-ID
Key-Id: 0X7da00001   USER-ID
Key-Id: 0X7fff0205   USER-ID
Key-Id: 0X7fff0209   COUNT                          Size(Bits): 32
Key-Id: 0Xf0000030   AES                            Size(Bits): 128
Key-Id: 0Xf0000003   BINARY                         Size(Bits): 3760
Key-Id: 0Xf0000001   BINARY                         Size(Bits): 3760
Key-Id: 0Xf0000002   NIST-P          (Key Pair)     Size(Bits): 256
Key-Id: 0Xf0000000   NIST-P          (Key Pair)     Size(Bits): 256
Key-Id: 0Xf0000012   NIST-P          (Key Pair)     Size(Bits): 256
Key-Id: 0Xf0000020   NIST-P          (Public Key)   Size(Bits): 256
Key-Id: 0X7fff0204   NIST-P          (Public Key)   Size(Bits): 256
Key-Id: 0X7fff0202   NIST-P          (Key Pair)     Size(Bits): 256
Key-Id: 0X7fff0201   NIST-P          (Key Pair)     Size(Bits): 256
Key-Id: 0X7fff0206   BINARY                         Size(Bits): 144
```

Figure 28. Connect to the EdgeLock SE05x using `ssscli` and read certificate ID list

4. We will now retrieve the device certificate from the EdgeLock SE05x.
   Send `>ssscli get cert 0xF0000001 device_cert.cer`. As shown in Figure 29, the certificate has been written to a file in the current path.
5. Finally, disconnect the communications to the EdgeLock SE05x. If the channel is not closed properly, you won't be able to establish a new connection until this command is executed.
   Send `>ssscli disconnect`.

```
C:\Windows\System32\cmd.exe
C:\se050_middleware\simw-top\binaries\pySSSCLI>ssscli get cert 0xF0000001 device_cert.cer
Getting Certificate from KeyID = 0xF0000001
Opening COM Port '\\.\COM9'
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.!!!
Retrieved Certificate from KeyID = 0xF0000001

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

Figure 29. Get the certificate from the EdgeLock SE05x using `ssscli`

3.6 Registering device certificate in AWS IoT Core

The next step is to register the device certificate in AWS IoT Core. For that, follow these steps:

1. From the AWS IoT Core dashboard, go to **Secure**, go to **Certificates** and click on the **Create a certificate** button as shown in Figure 30:

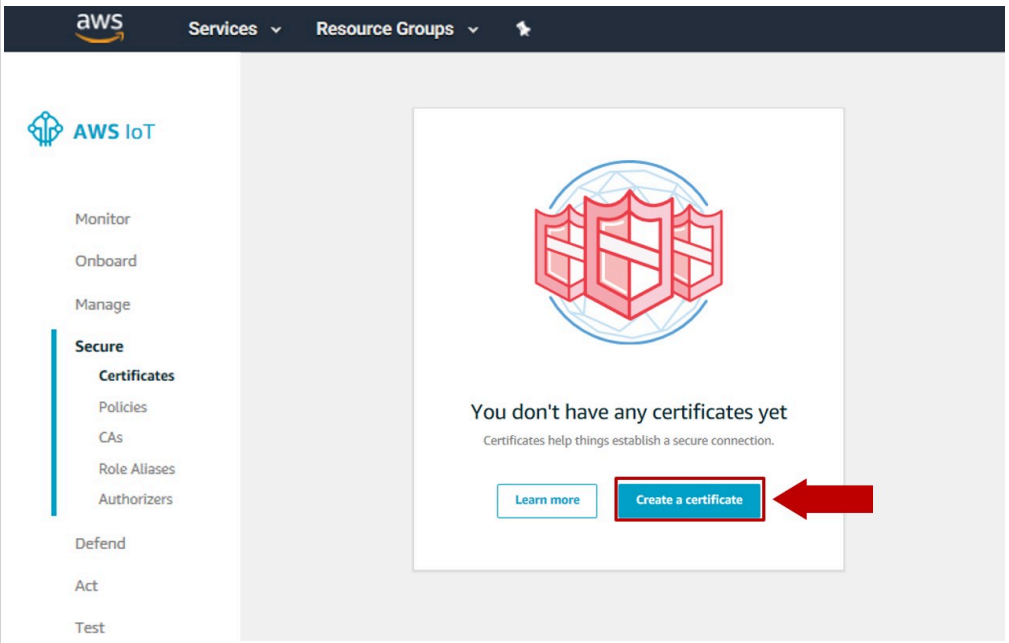

Figure 30. Go to the AWS IoT Certificates menu

2. A new menu called *Create a certificate* will be opened. Click on the **Get started** option as shown in Figure 31:

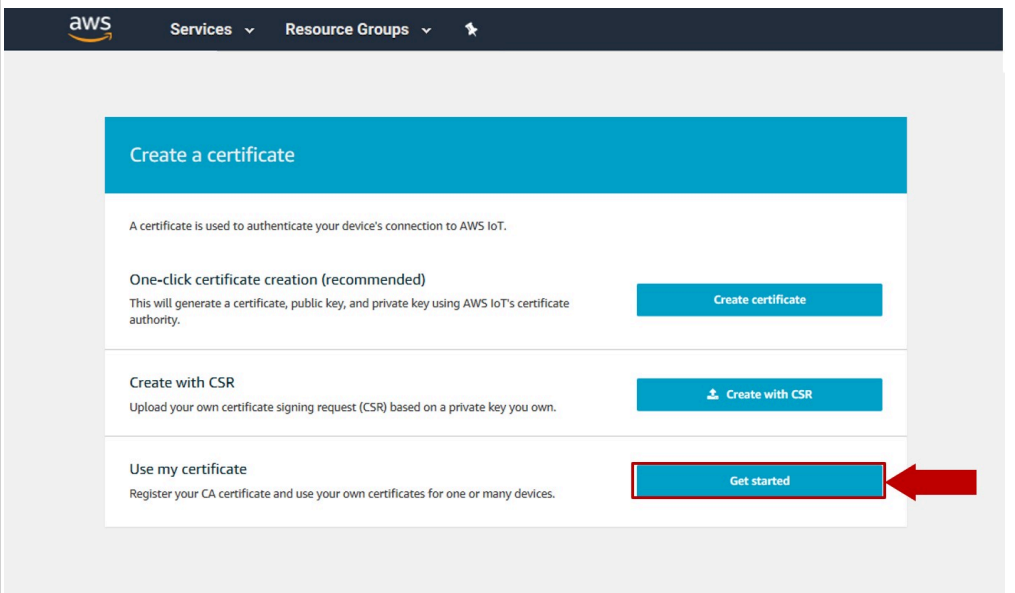

Figure 31. Register a certificate

A new menu called *Select a CA* will be opened. In this menu, choose ***Next*** as shown in Figure 32:

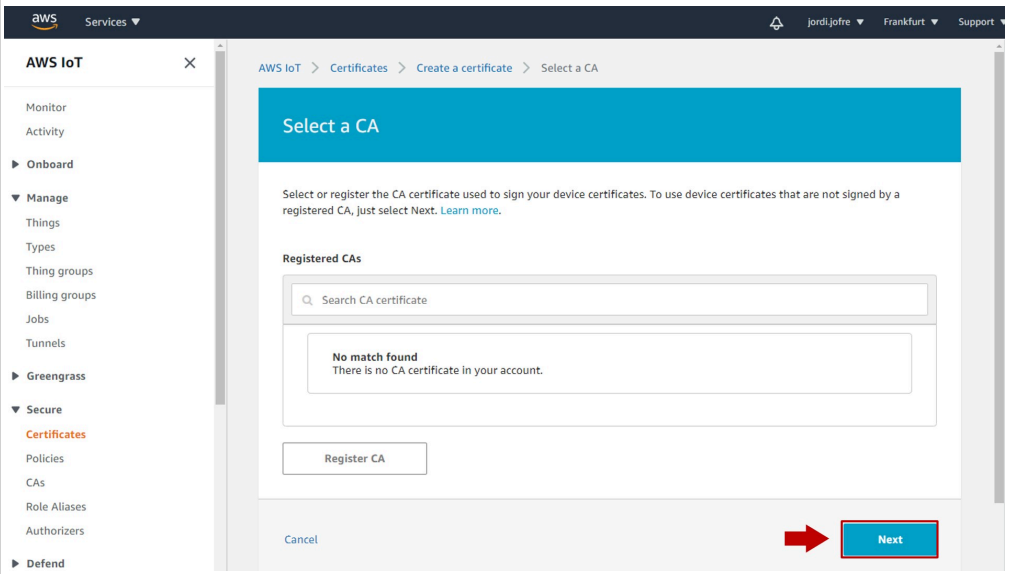

Figure 32. Select CA menu

3. On ***Register existing device certificates***, choose ***Select certificates***, and select the certificate exported from EdgeLock SE05x in Section 3.5, as shown in Figure 33:

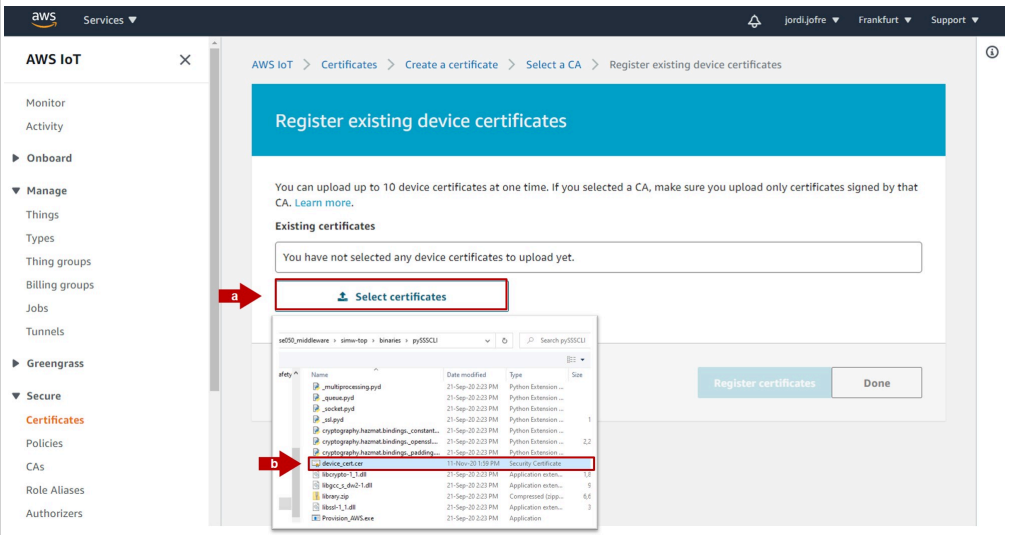

Figure 33. Register existing device certificate

4. After closing the file dialog box, select ***Activate all*** and then click on ***Register certificates***, as shown in Figure 34:

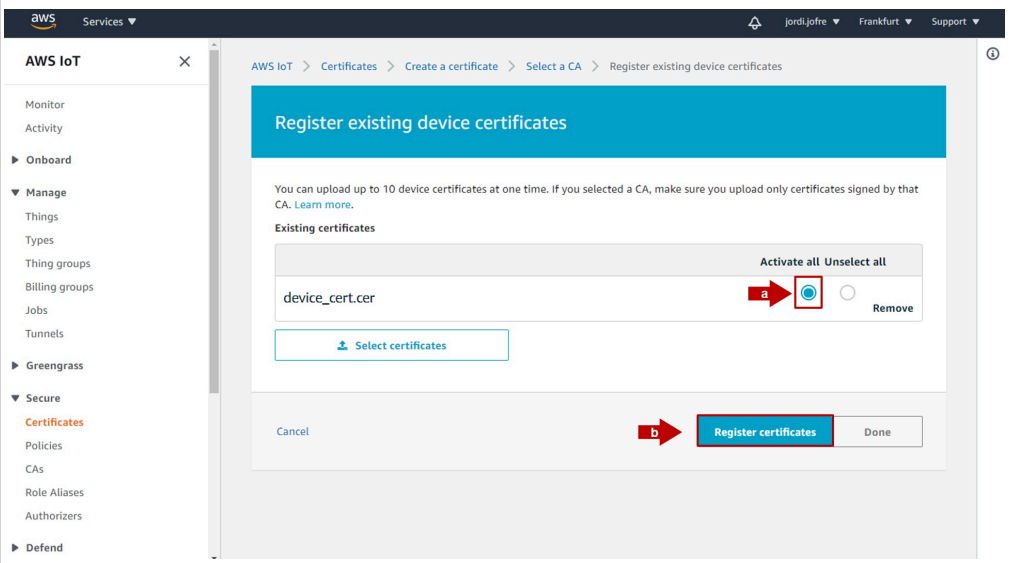

Figure 34. Activate device certificate

The device certificates that are registered successfully appear in the list of certificates.

3.7 Attach AWS Thing and policy to the certificate

Finally, we only need to attach the thing and policy we created back in Section 3.3 and Section 3.4, respectively, to the newly registered device certificate. Go to the AWS IoT Core administration console and follow the steps:

1. Attach your thing to the certificate following the instructions shown in Figure 35.
   a. Click on the top right corner to go to the device certificate options.
   b. Click on **Attach a thing**.
   c. Select the AWS IoT Thing you created in Section 3.3. In this example, it was called **my_thing**.
   d. Click on the **Attach** button.

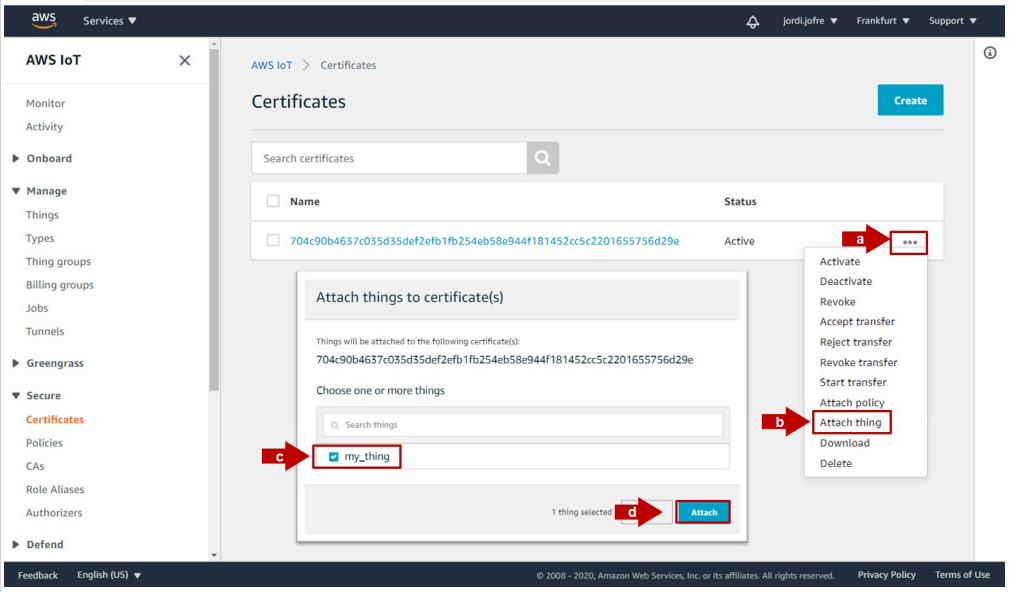

Figure 35. Attach a thing to your device certificate

2. Attach your policy to the certificate as shown in Figure 36.
   a. Click on the top right corner to go to the device certificate options.
   b. Click on **Attach a policy**.
   c. Select the AWS IoT Policy created in Section 3.4. In this example, it was called **my_policy**.
   d. Click on the **Attach** button.

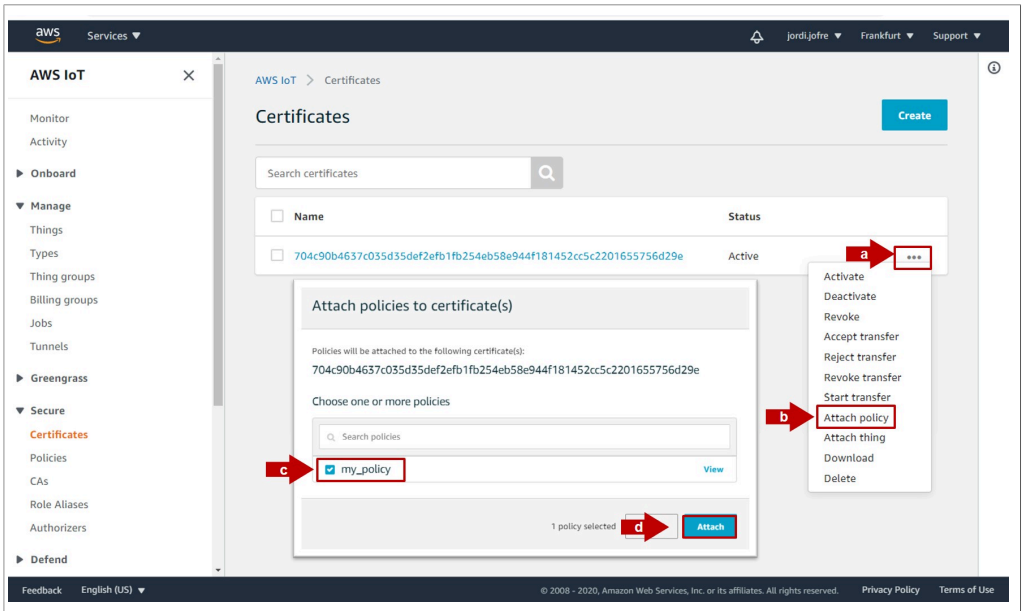

Figure 36. Attach a policy to your device certificate

3.8 AWS IoT Core project configuration

To run the AWS project example using the FRDM-K64F board, we need to:

- Download and install the FRDM-K64F SDK
- Import AWS IoT Core example project
- Configure AWS IoT Core project account settings
- Execute AWS IoT Core example project

Note: *Before running the AWS IoT Core demo example, you need to have installed MCUXpresso IDE and FRDM-K64F SDK in your local environment and imported the AWS IoT Core 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 AWS IoT Core project example.*

3.8.1 Download and install the FRDM-K64F SDK

The AWS IoT Core device onboarding project example is included as part of the FRDM-K64F SDK. Install it to your MCUXpresso workspace as shown in Figure 37:

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.

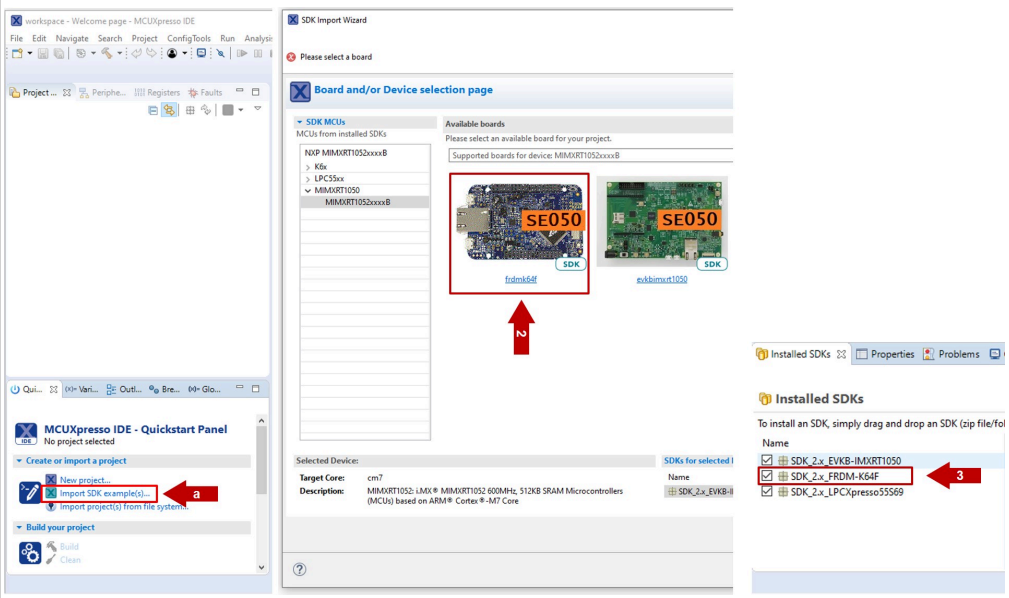

Figure 37. Import FRDM-K64F SDK

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.

3.8.2 Import AWS IoT Core example project

The FRDM-K64F SDK includes a project example called `se_SE050x_cloud_aws`. Import it to your MCUXpresso workspace as shown in Figure 38:

1. Click *Import SDK examples* from the MCUXpresso IDE quick start panel.
2. Select `se_SE050x_cloud_aws` 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

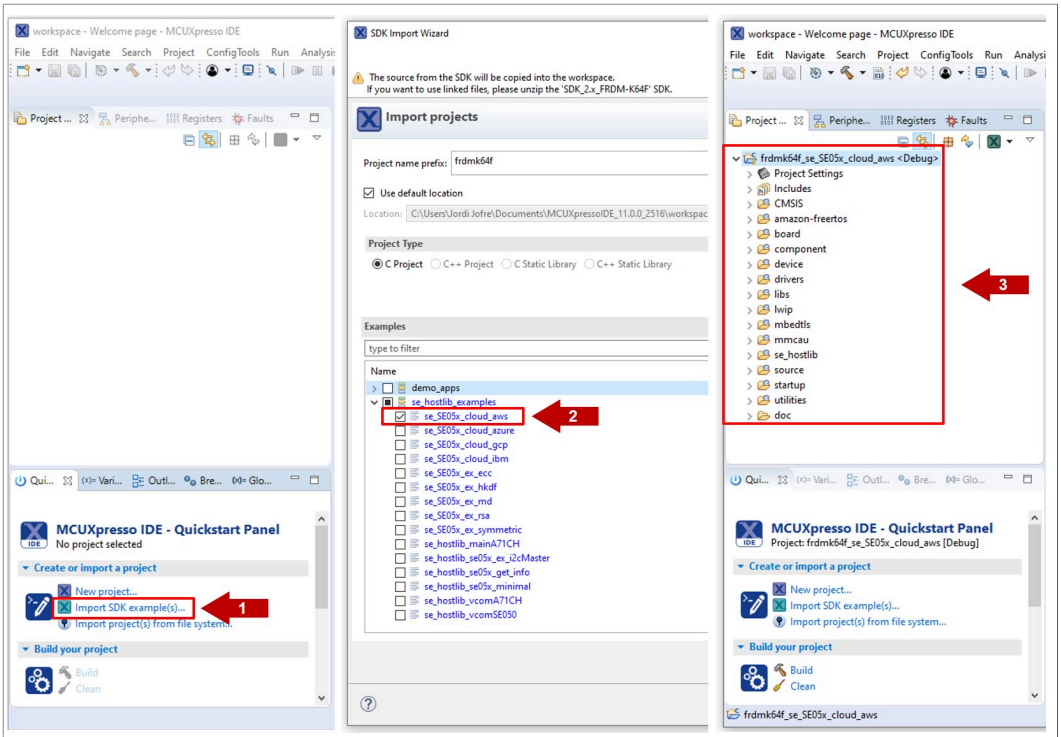

Figure 38. Import AWS project in the workspace

3.8.3 Configure AWS IoT Core project account settings

We need to change the AWS Rest API Endpoint in the MCUXpresso demo project with the one in your AWS IoT Core account settings. Follow these steps:

1. From the AWS IoT Core dashboard, go to *Manage*, then go to *Things* and click on your AWS IoT Thing as shown in Figure 39:

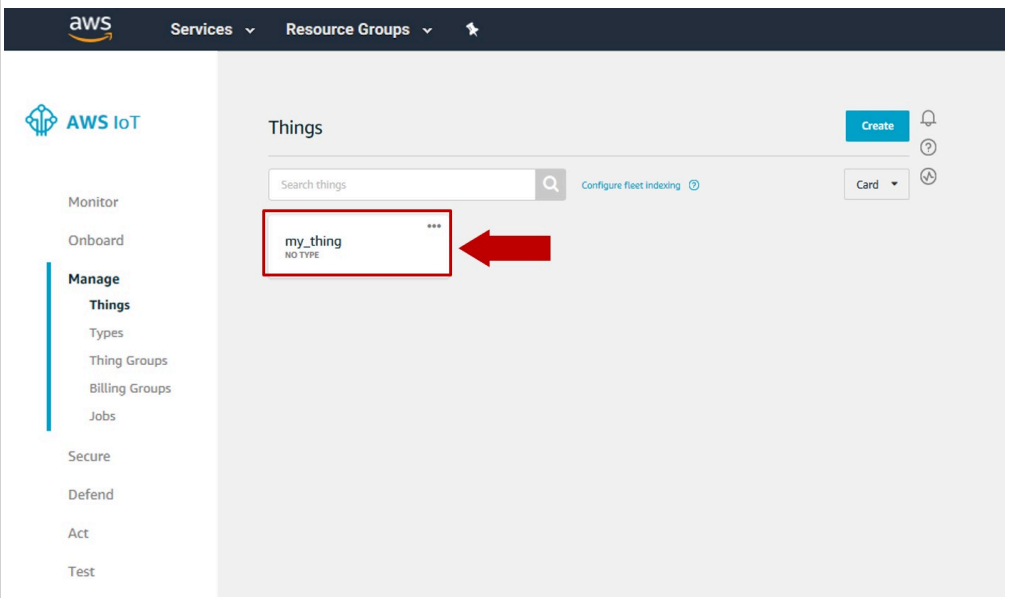

Figure 39. Go to your AWS IoT Thing

2. On the left hand side menu, (1) go to **Interact**. Inside this menu, you will find your (2) Rest API Endpoint as indicated in Figure 40. Copy this URL.

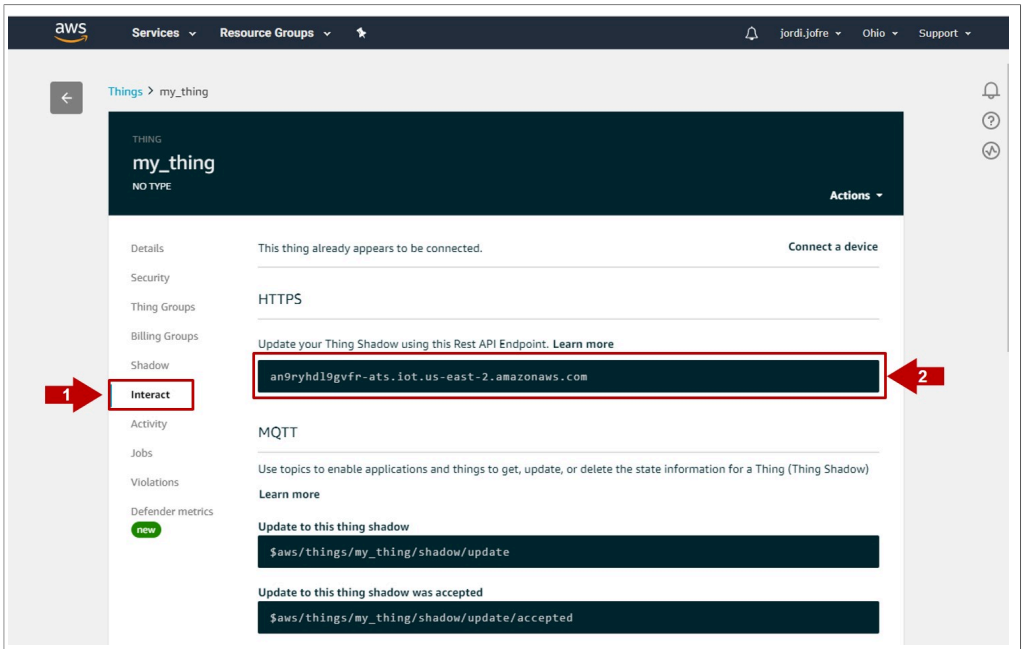

Figure 40. Find your Rest API Endpoint

3. Go to the AWS demo in your MCUXpresso workspace. Navigate to the `aws_clientcredential.h` file located in `frdmk64f_se_SE05x_cloud_aws\source` folder. Replace the `clientcredentialMQTT_BROKER_ENDPOINT`

variable with the Rest API Endpoint of your AWS account obtained in the previous step, as well as your thing name as created in Section 3.3. Check Figure 41 for reference.

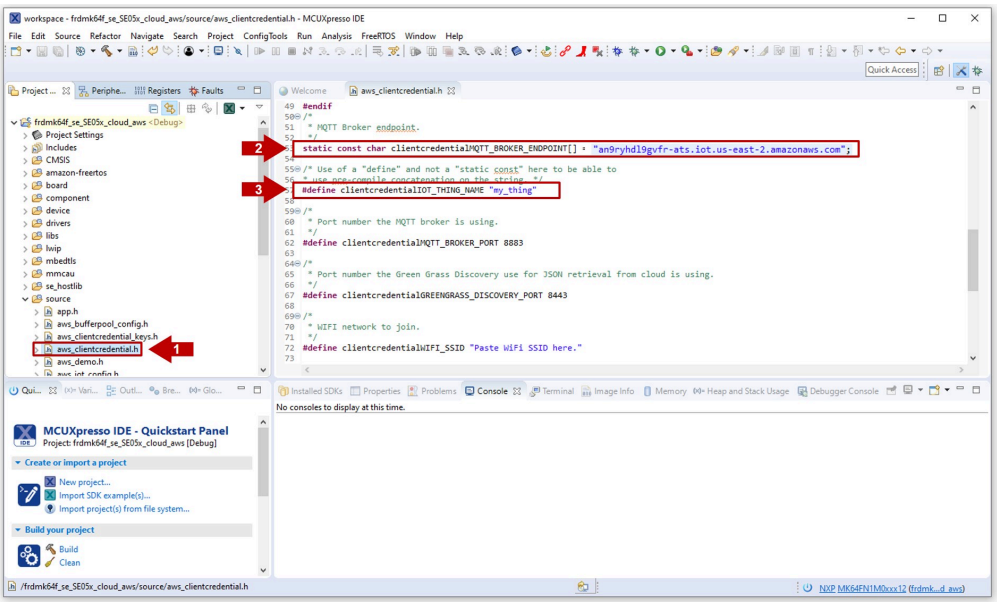

Figure 41. Set the Rest API Endpoint and thing name in your project settings

4. On the same **Interact** menu, you will find MQTT topics that enable applications and things to get, update, or delete the state information for an AWS thing. For instance, copy the MQTT ***update*** topic as shown in Figure 42:

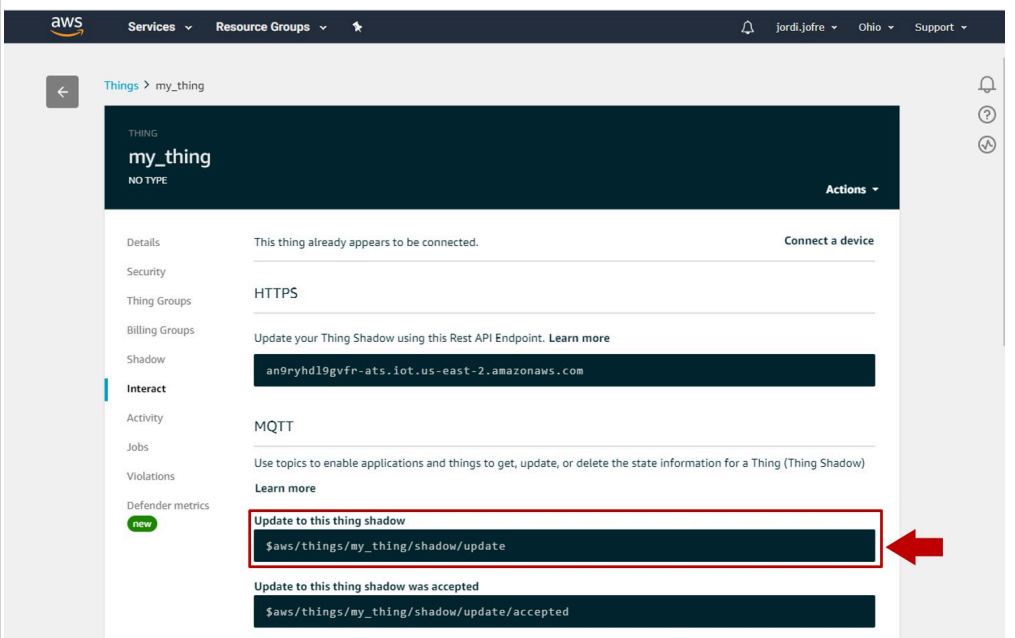

Figure 42. Find your MQTT *Update* topic

5. Go to the AWS demo in your MCUXpresso workspace. Navigate to the `aws_jitr_task_lwip.c` file located in `frdmk64f_se_SE05x_cloud_aws\source` folder. Replace the `#define PUB_TOPIC` variable with the MQTT topic you obtained in Figure 41 as shown in Figure 42.

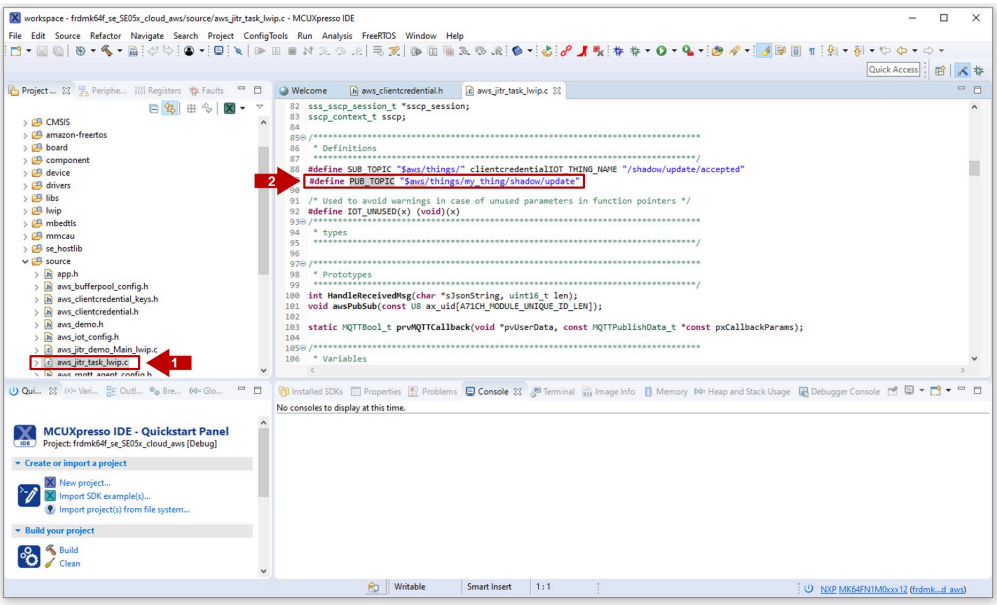

Figure 43. Set the MQTT publish topic

6. Finally, we need to tell the FRDM-K64F board which credentials to use. Recall from Section 3.5 that we are using key ID `0xF0000000` and certificate ID `0xF0000001`. Therefore, navigate to the `aws_iot_config.h` file located in

`frdmk64f_se_SE05x_cloud_aws\source` folder and set the `#define` lines accordingly as shown in Figure 44.

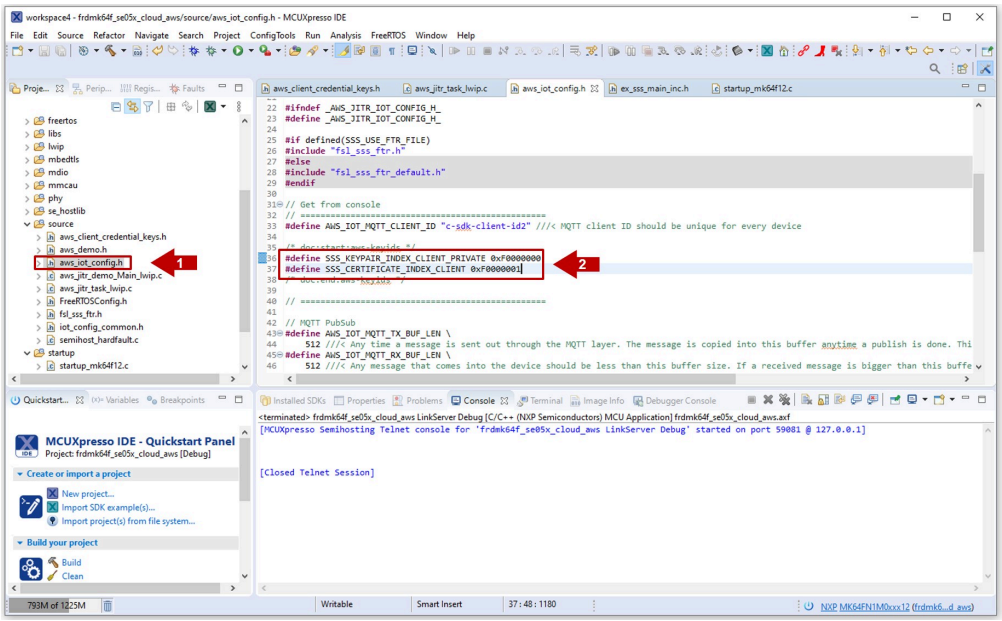

Figure 44. Setting the credential IDs

Everything is now fully ready to run the demo on the FRDM-K64F board. Please jump back to Section 3.9 to execute the demo project and verify that everything is running as expected.

3.9 AWS IoT Core project execution

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

1. Subscribe to the MQTT topic from Figure 87. Go to the AWS IoT Core dashboard and follow the steps indicated in Figure 45:
   a. Go to ***Test***.
   b. Go to ***Subscribe to a topic***.
   c. Write the MQTT topic name in the Subscription topic field.
   d. Click on ***Subscribe to topic*** button.

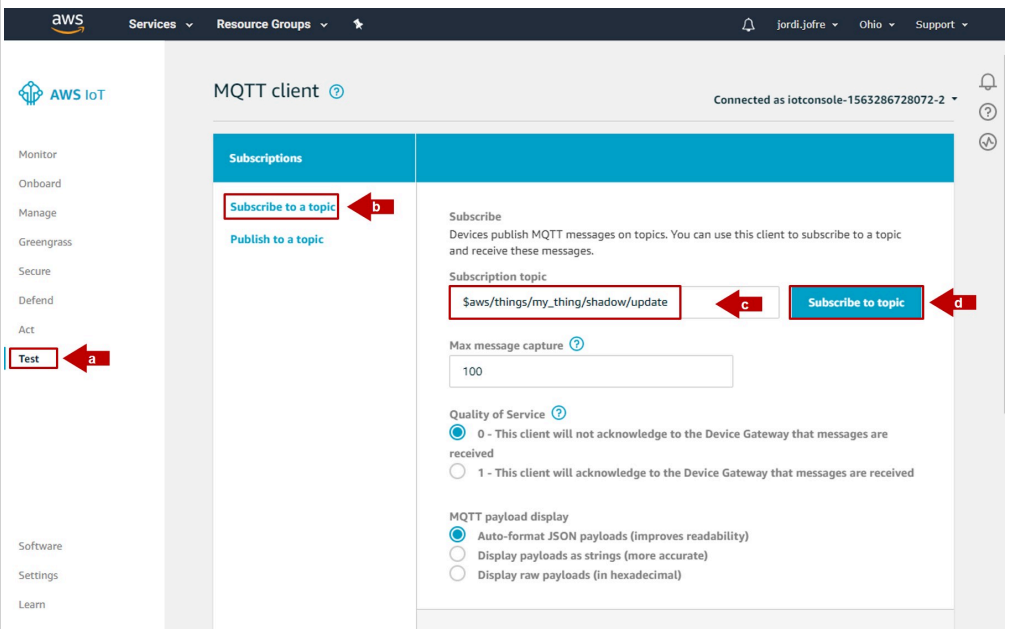

Figure 45. Subscribe to the MQTT topic

2. The MQTT topic you subscribed will now appear in the ***Subscriptions*** section as shown in Figure 46:

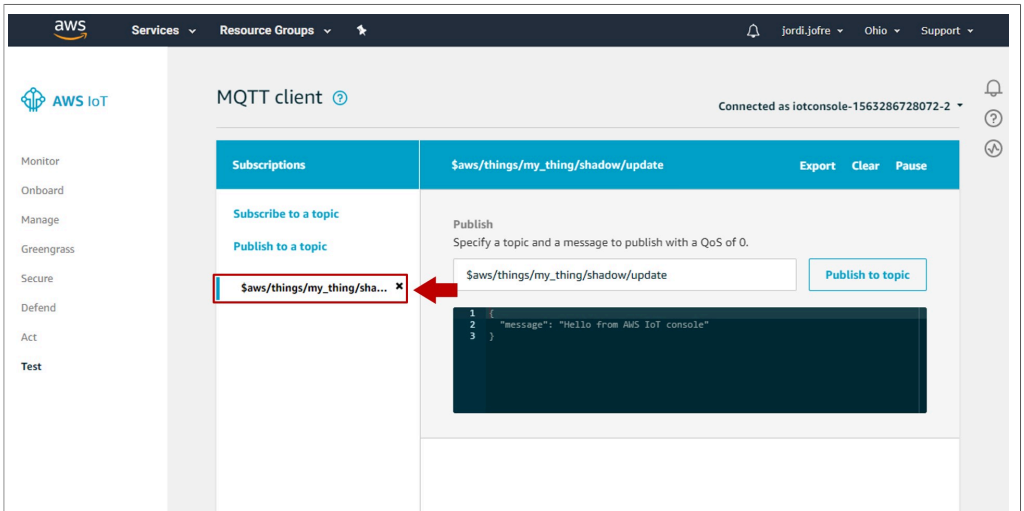

Figure 46. Check MQTT topic subscription

3. Connect FRDM-K64F OpenSDA port, K64F port and Ethernet interface to your laptop as shown in Figure 47:

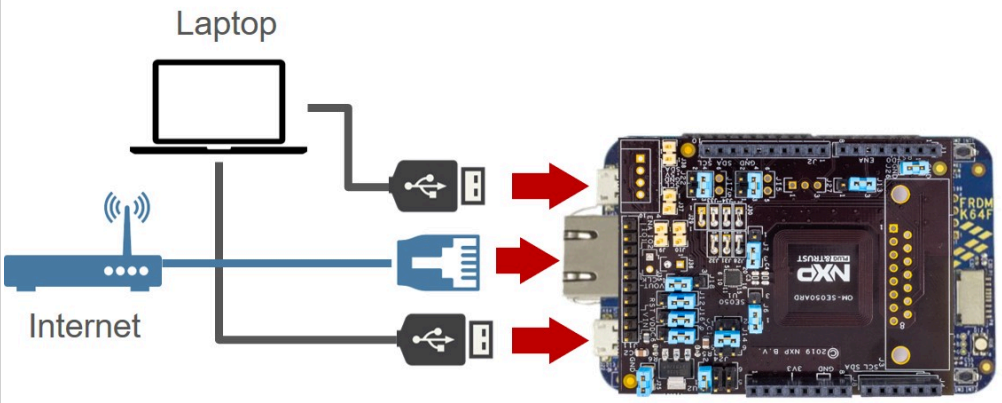

Figure 47. Connect FRDM-K64F board

4. 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.

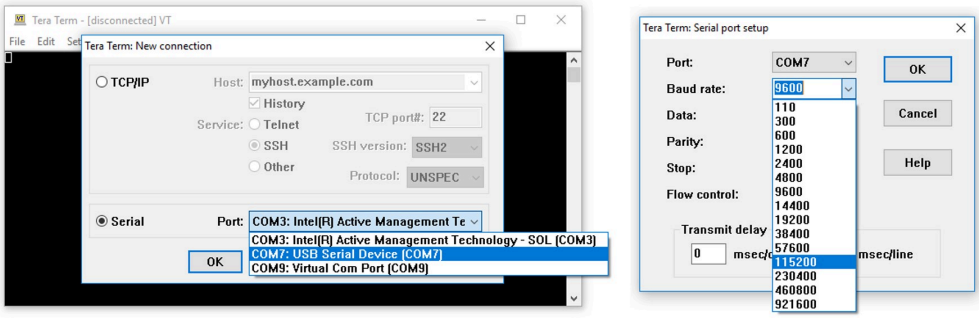

Figure 48. Configure TeraTerm

5. 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 49:

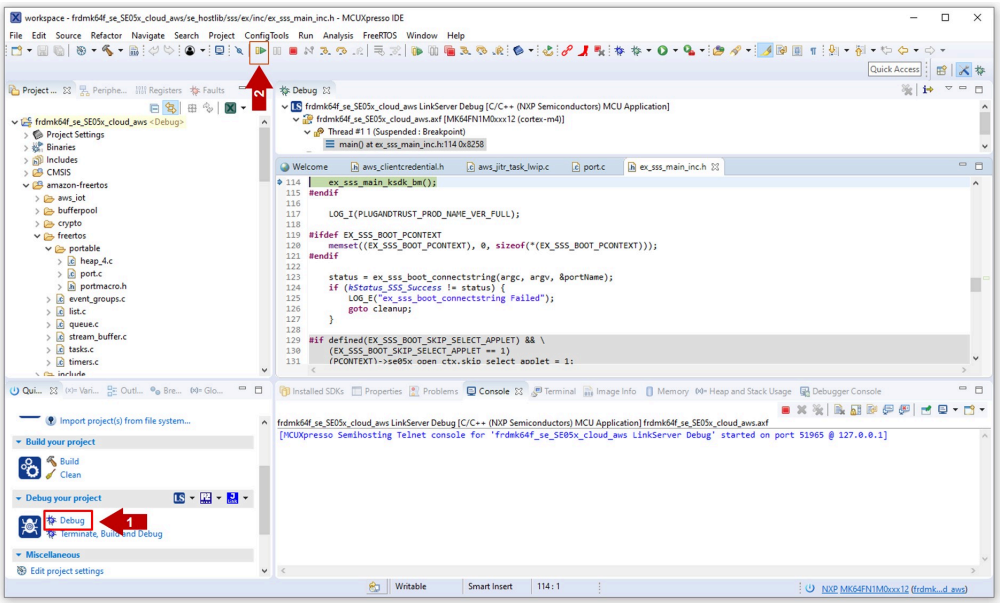

Figure 49. Debug AWS project

6. Your device should now be connected to AWS. Check that your device is connected by:
   a. Checking the TeraTerm logs as shown in Figure 50.

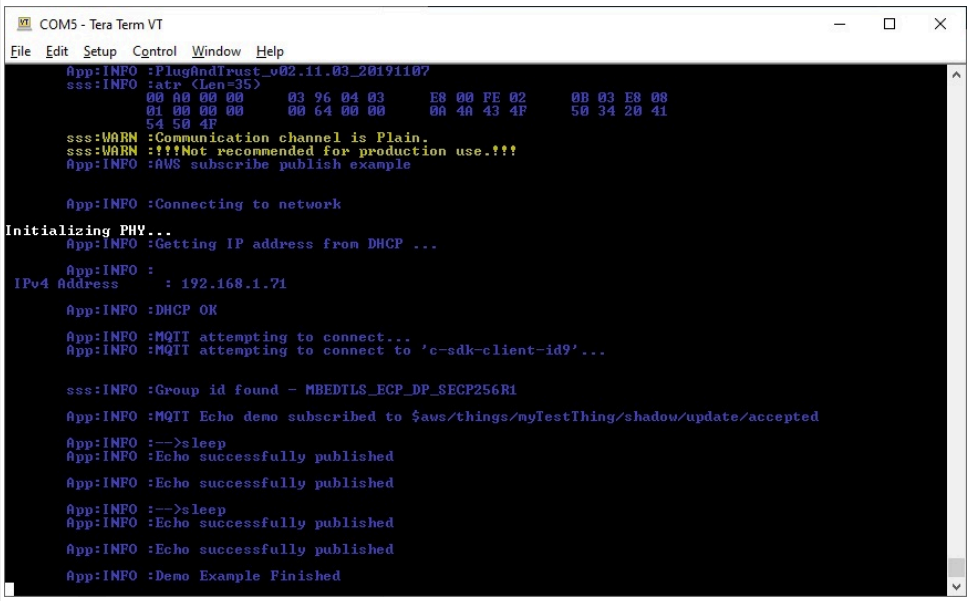

Figure 50. Device connection to AWS

   b. Checking the last time the device was seen in the AWS dashboard as shown in Figure 51.

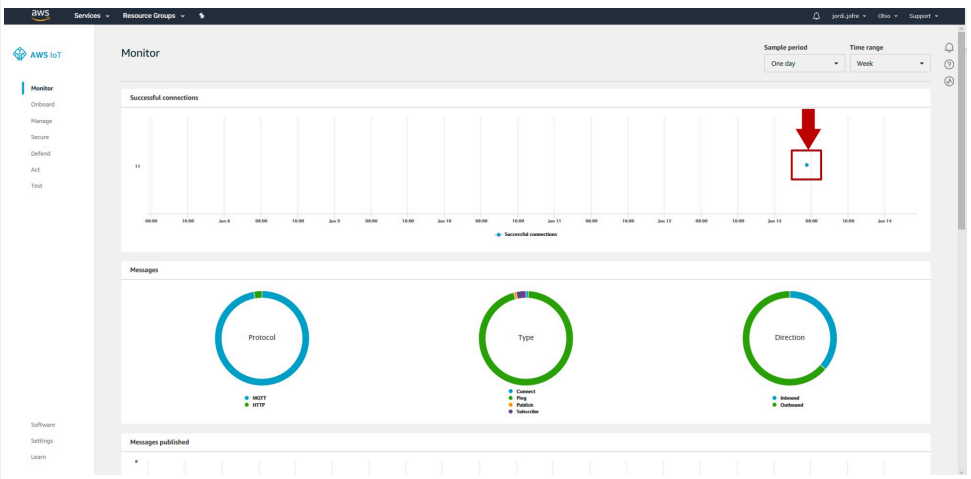

Figure 51. Device connection to AWS - dashboard

   c. Checking the messages published in the subscribed MQTT topic as shown in Figure 52:

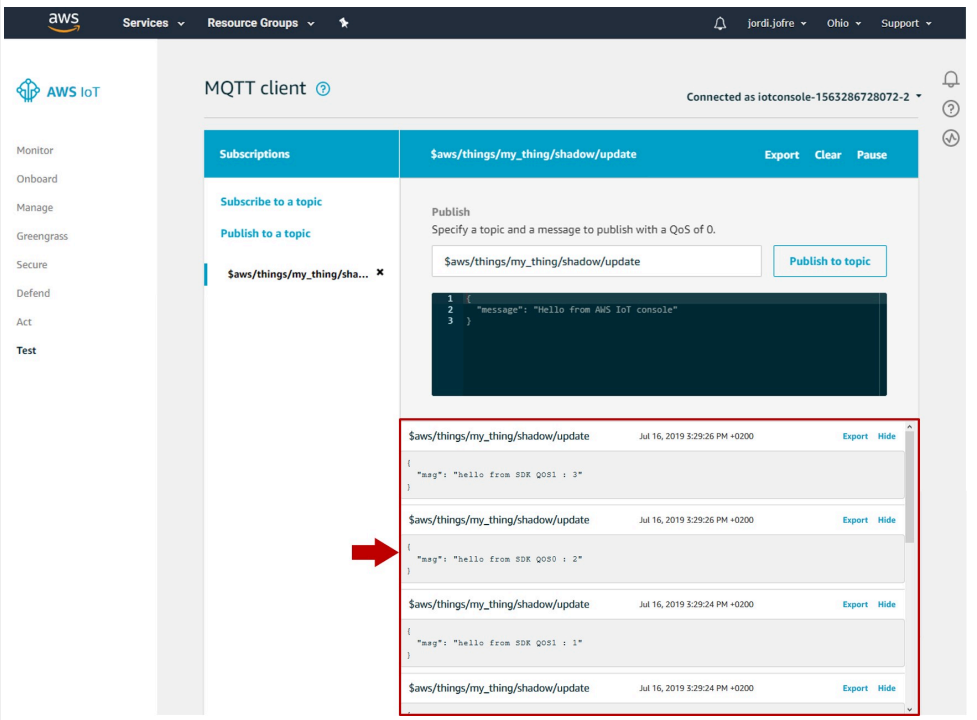

Figure 52. Device connection to AWS - Published messages in the *update* MQTT topic

4 Appendix: Registering a CA certificate for just-in-time registration

Alternatively to the procedure explained in Section 3, you can configure a CA certificate to enable device certificates it has signed to register with AWS IoT automatically the first time the device connects to AWS IoT. To register device certificates when a client connects to AWS IoT 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 Running AWS IoT Core key provisioning scripts

This section explains how to generate the credentials for the EdgeLock SE05x using the key provisioning scripts 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 AWS IoT Core.

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

4.1.1 Download EdgeLock SE05x Plug & Trust Middleware

Follow these steps to download the EdgeLock SE05x Plug & Trust Middleware in your local machine:

1. Download EdgeLock SE05x Plug & Trust Middleware from the NXP website.

2. Create a folder called ***se050_middleware*** in C: directory as shown in Figure 53:

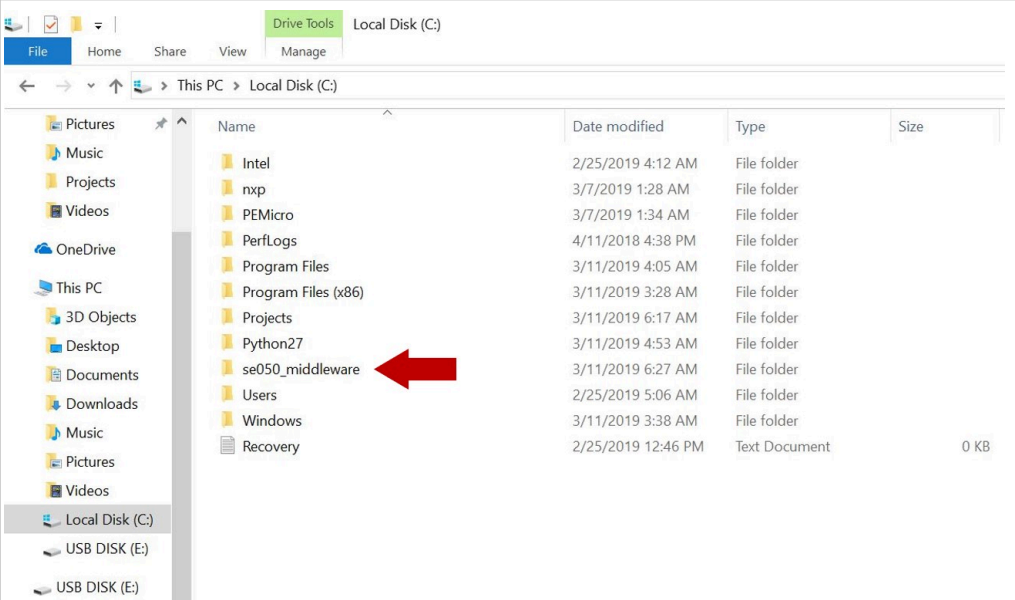

Figure 53. Create se050_middleware folder

3. Unzip the EdgeLock SE05x Plug & Trust Middleware inside the *se050_middleware* folder. After unzipping, you will see a folder called ***simw-top*** created. The contents of the ***simw-top*** directory should look as shown in Figure 54:

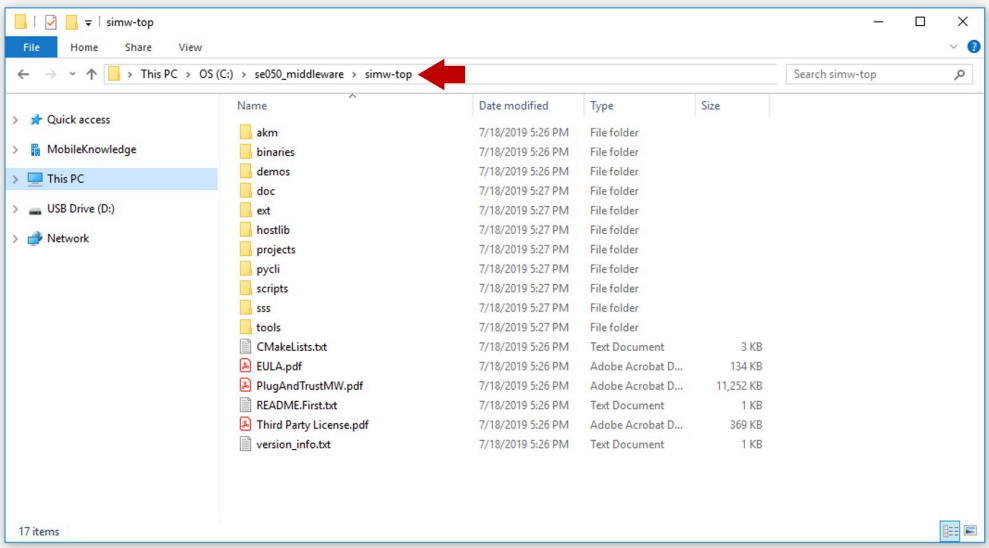

Figure 54. Unzip se050 middleware

Note: *It is recommended to keep* `se050_middleware` *with the* ***shortest*** *path possible and* ***without spaces*** *in it. This avoids some issues that could appear when building the middleware if the path contains spaces.*

4.1.2 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 55:

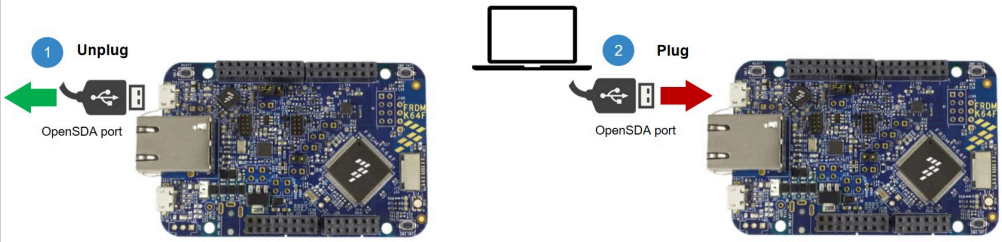

Figure 55. Unplug and plug OpenSDA port

2. When you plug the board, your laptop should recognize the board as an external drive as shown in Figure 56:

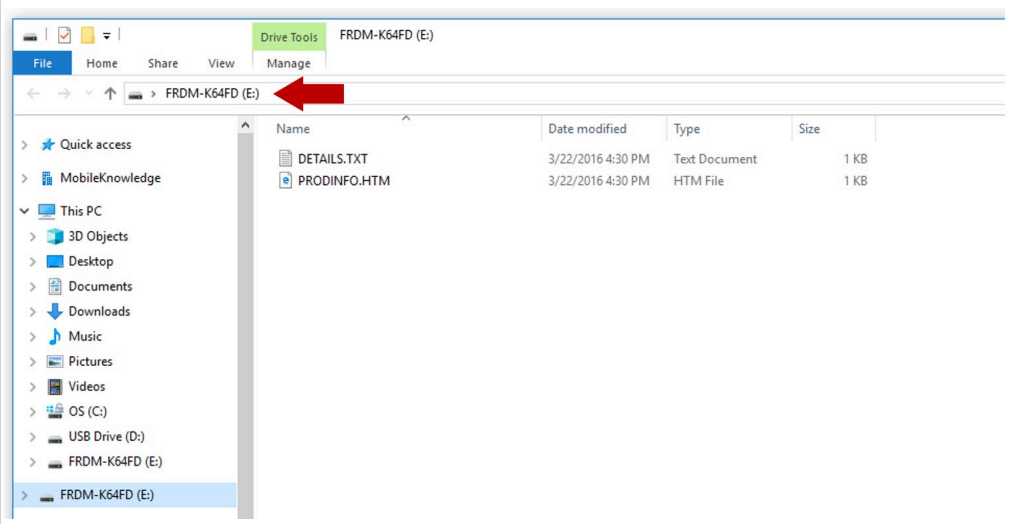

Figure 56. FRDM-K64F drive

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 57:

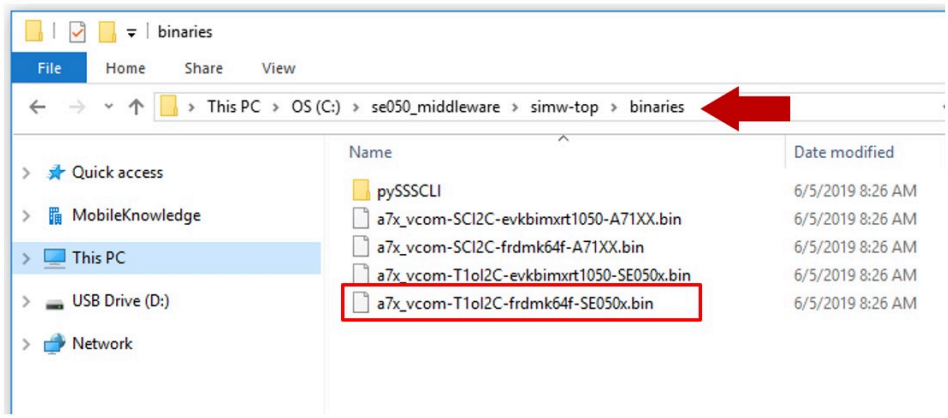

Figure 57. VCOM binary folder

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 58:

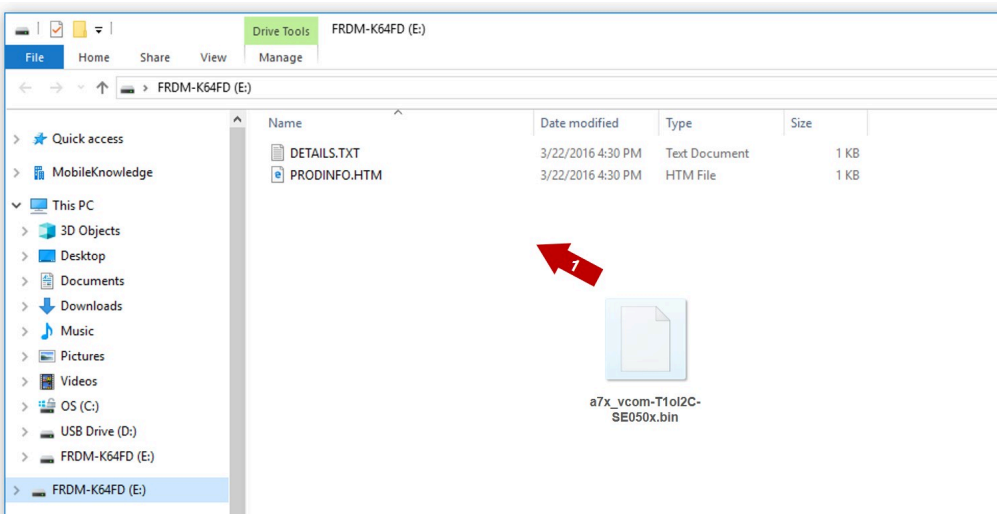

Figure 58. Drag and drop VCOM binary

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 59:
   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.

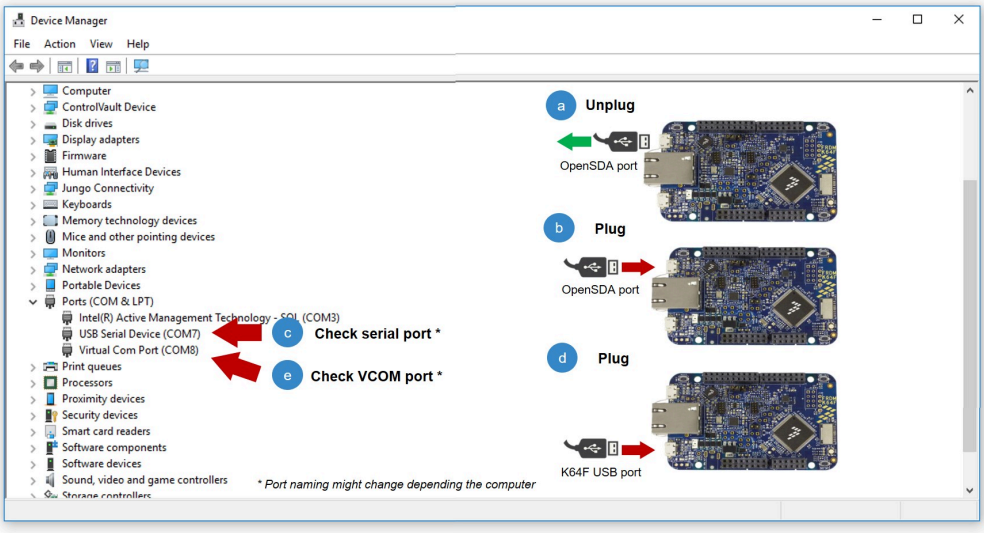

Figure 59. Check VCOM and serial ports

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.*

4.1.3 Key and certificate configuration for use with AWS IoT Core

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 AWS IoT Core demo.

On the other hand, it is also possible to use the pre-provisioned credentials that are already in the EdgeLock SE05x for this purpose thanks to the Ease of Use configuration. However, this method requires an AWS feature called 'Multi-Account Registration'. If you wish to configure the credentials with the Ease of Use configuration, please skip ahead to Section 3.5.

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 60

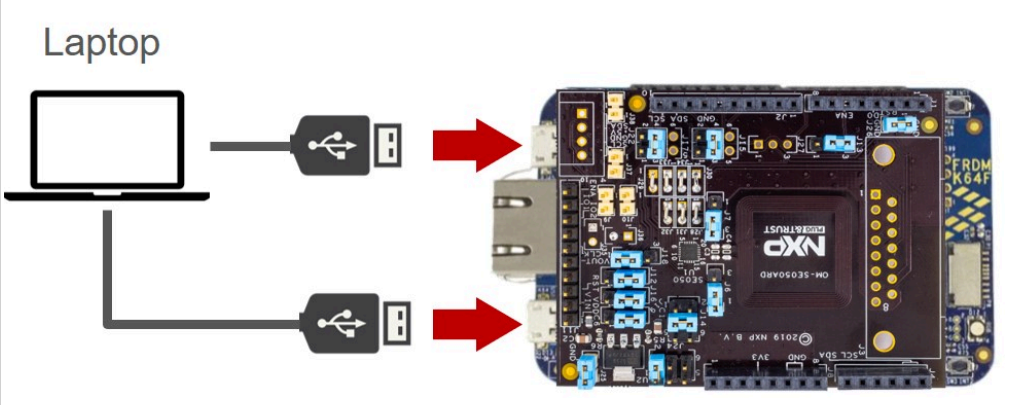

Figure 60. Connect boards

2. Go to `simw-top\binaries\pySSSCLI` folder and locate the `Provision_AWS.exe` file as shown in Figure 61:

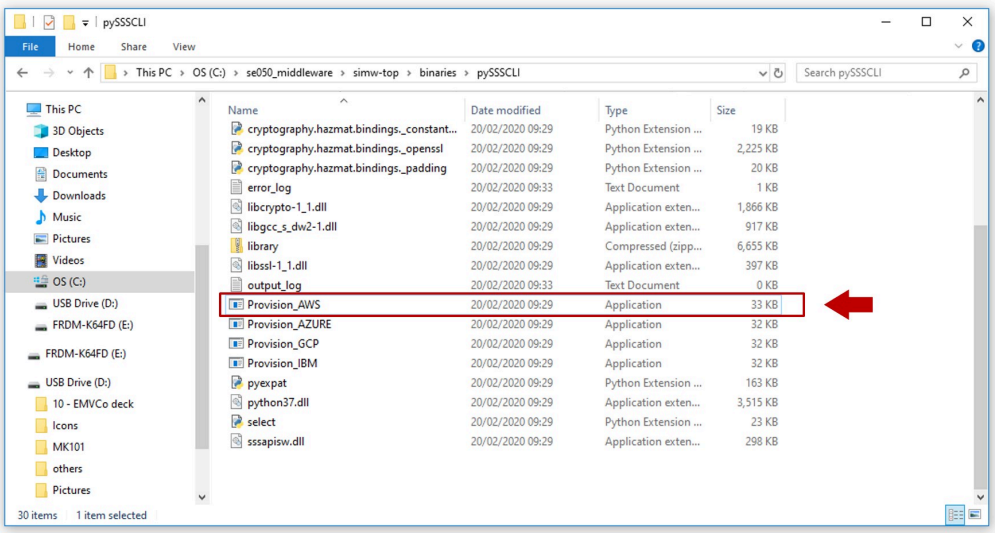

Figure 61. Find Provision_AWS.exe file in your EdgeLock SE05x Plug & Trust Middleware package

3. Open a command prompt

4. Use the `Provision_AWS.exe` executable to generate and inject keys into your EdgeLock SE05x. You cal follow these steps shown in Figure 62:
   a. Go to the folder `simw-top\binaries\pySSCLI` and run `>cd C:\se050_middleware\simw-top\binaries\pySSSCLI`
   b. Run the executable `Provision_AWS.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_AWS.exe COM8`
   c. Check that the keys are generated and injected.
   d. Check that the program execution completes successfully.

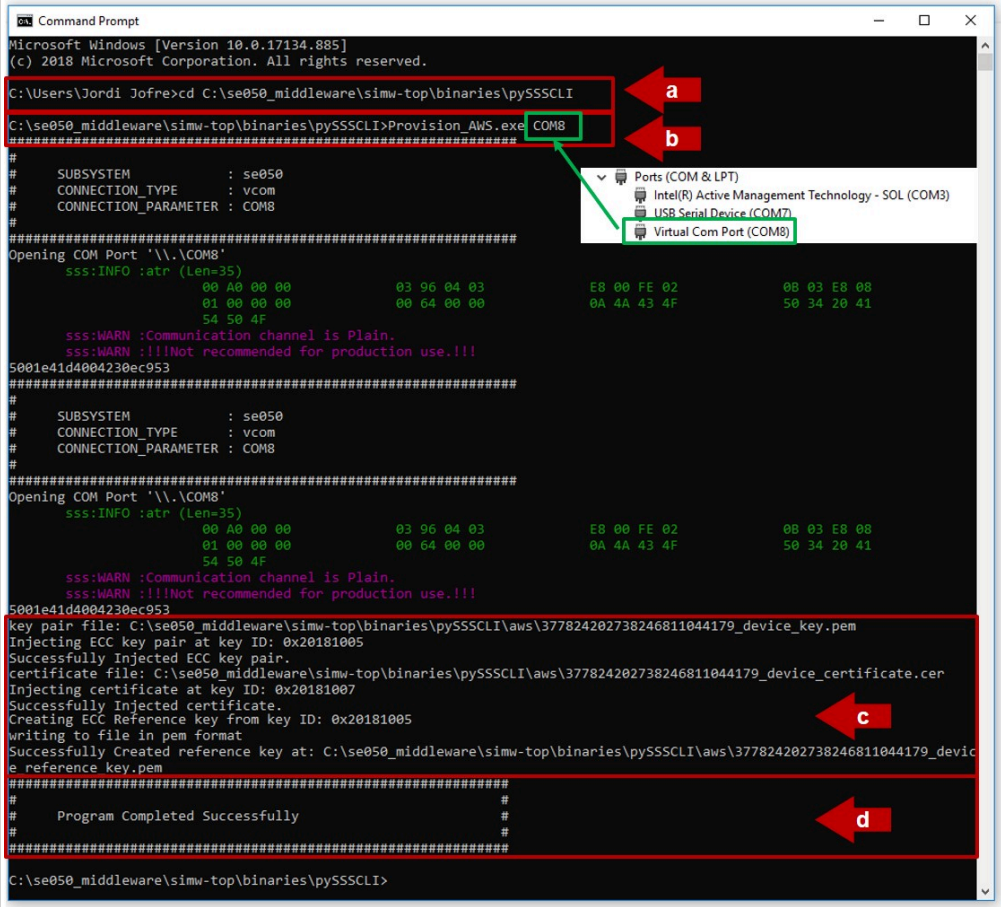

Figure 62. Run `Provision_AWS.exe` executable

5. Go to `simw-top\binaries\pySSCLI\aws` folder and check that the keys appear inside the folder as shown in Figure 63:

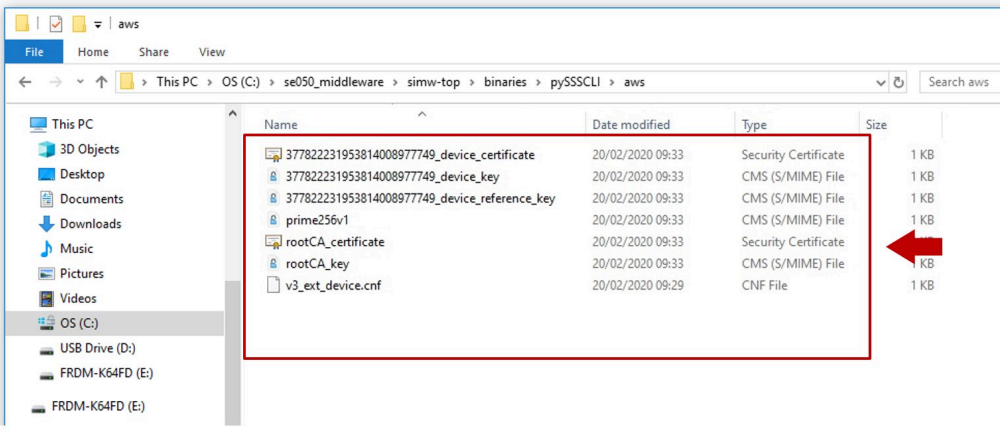

Figure 63. Generated AWS credentials

After injecting the credentials, go to Section 4.2

4.2 Register root certificate authority (CA)

This section describes how to register the root CA certificate with AWS IoT Management Console. For the sake of simplicity, this application note only uses the AWS IoT Management Console web interface. For details on how to perform any of these steps using other tools provided by AWS, refer to AWS Core IoT documentation.

Note: *The AWS IoT Core account preparation procedure is the same independently of the MCU / MPU platform you choose for evaluation purposes.*

4.2.1 Get registration code from AWS

AWS IoT Core requires the registration of a CA certificate used to sign and issue your device certificates. This CA certificate is used for authentication of devices attempting to connect to the platform thereafter. As part of the CA certificate registration process, AWS IoT Core performs a *proof-of-possession* verification. This *proof-of-possession* mechanism ensures that the uploader of the CA certificate also knows the associated private key. The *proof-of-possession* mechanism consists of generating a *verification certificate* using:

- The *CA certificate*
- The *CA private key*
- A *registration code* given by AWS.

To generate the verification certificate, the AWS IoT Core registration code needs to be set in the Common Name field of the verification certificate signed by the CA certificate private key.

As a first step, we need to obtain the AWS *registration code* assigned to our account. For the sake of simplicity, this application note only uses the AWS IoT Management Console web interface. For details on how to perform any of these steps using other tools

provided by AWS, refer to AWS Core IoT documentation.To obtain AWS registration code follow these steps:

1. On the menu of the left hand side of the AWS IoT Core dashboard, go to ***Secure***, select ***Certificates*** and click ***Create a certificate*** as shown in Figure 64:

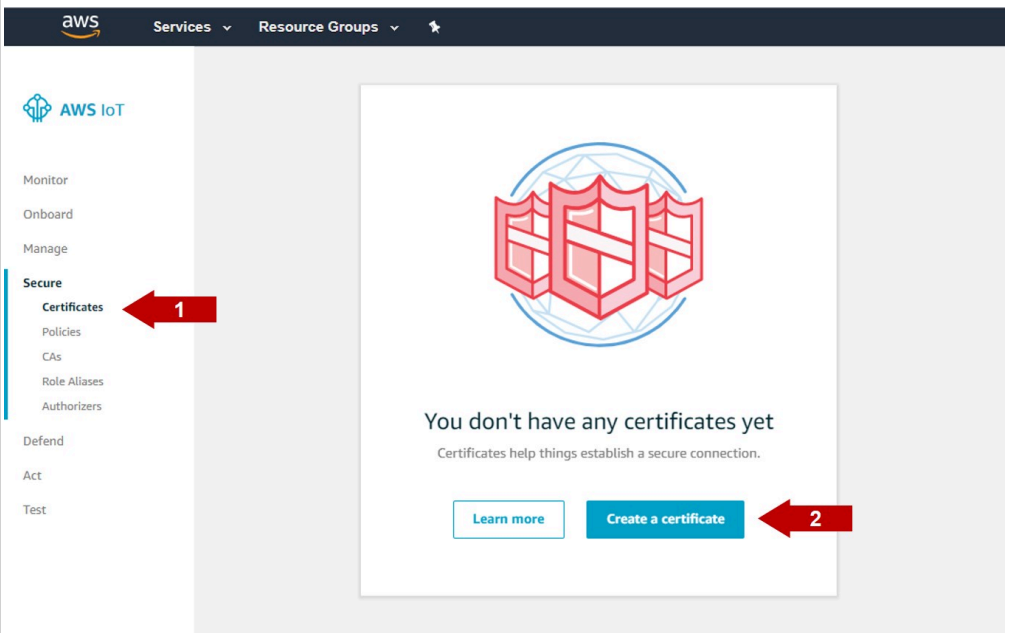

Figure 64. AWS IoT certificates menu

AWs IoT Core supports three options:

- ***One-click certificate creation***: AWS IoT Core generates an individual certificate and their associated public and private keys for a device.
- ***Create a CSR***: OEM generates a device key pair and generates a certificate signing request (CSR). This CSR is signed by AWS IoT Core certificate authority (CA).
- ***Use my certificate***: OEM generates the device certificates. In this option, the OEM needs to register the CA certificate that signed and issued the device certificates.

This document describes the *Use my certificate* option.

2. Select the ***Use my certificate*** option as shown in Figure 65:

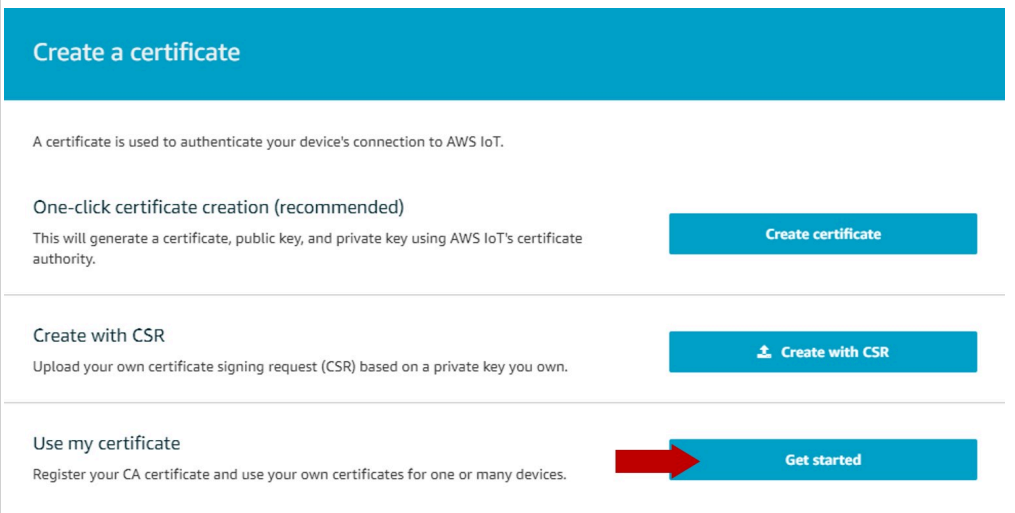

Figure 65. AWS certificate creation options

3. Click ***Register CA button*** as shown in Figure 66:

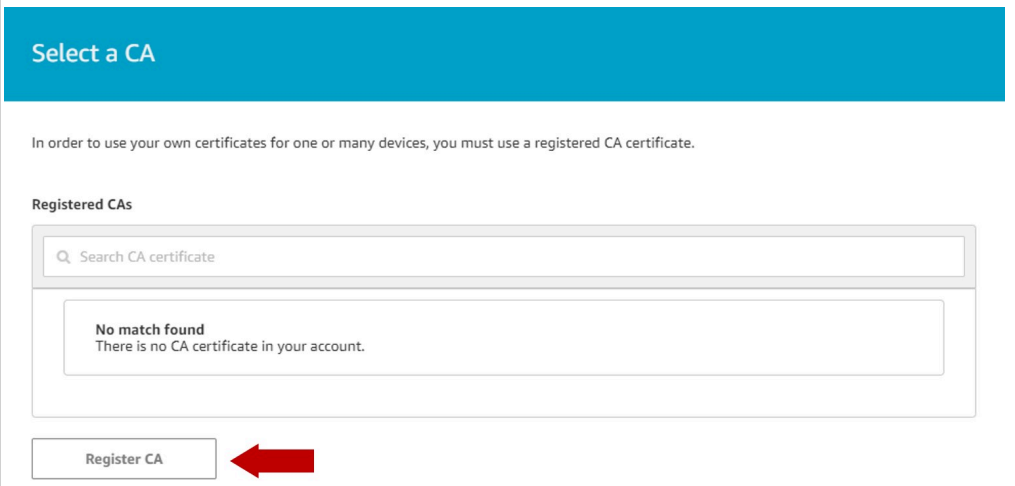

Figure 66. Register a CA certificate

4. You will see a form with instructions to register a CA certificate. Go to step 2, and copy the **registration code** as shown in Figure 67 for later use.

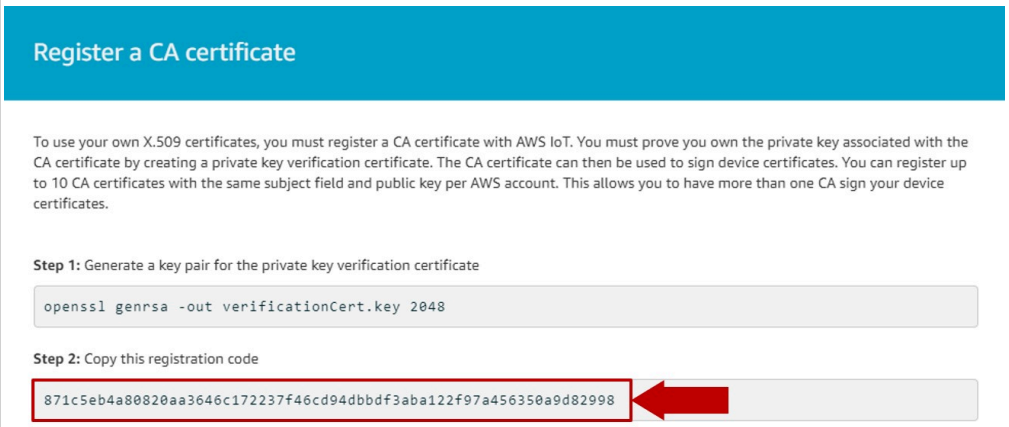

Figure 67. Get registration code

4.2.2 Generate AWS verification certificate

The EdgeLock SE05x Plug & Trust Middleware includes a Python script called `verification_certificate.py` that generates the AWS verification certificate. This scripts needs three arguments:

- The path to the root CA certificate
- The path to the root CA private key
- The AWS registration code

To generate the AWS verification certificate, follow these steps:

1. Go to `C:\se050_middleware\simw-top\pycli\Provisioning` folder and find the file `verification_certificate.py` as shown in Figure 68:

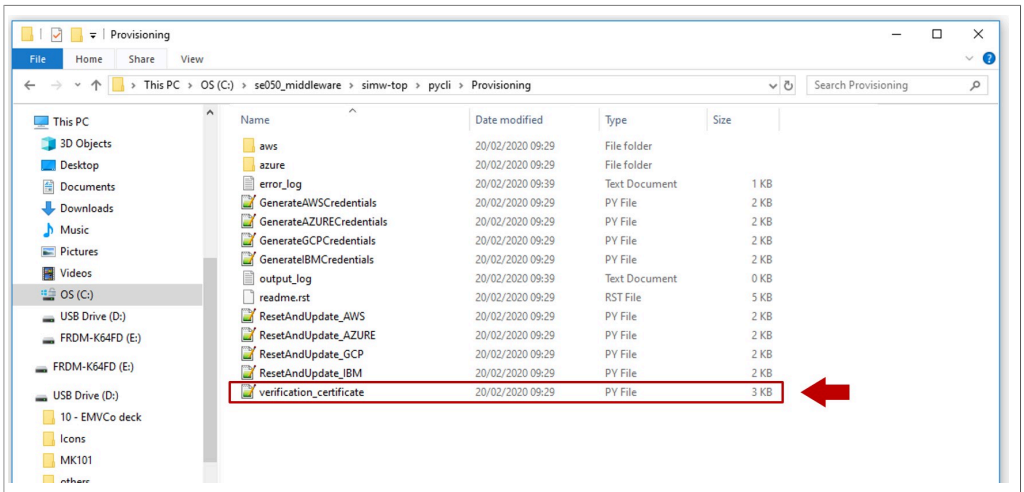

Figure 68. Locate `verification_certificate.py` Python script

2. Open a Command prompt

3. Go to `C:\se050_middleware\simw-top\pycli\Provisioning` folder as shown in Figure 69.
   Send >`cd C:\se050_middleware\simw-top\pycli\Provisioning`

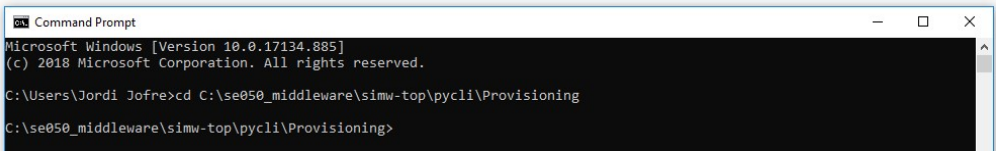

Figure 69. Go to Provisioning folder

4. Execute the `verification_certificate.py` Python script. (Figure 70) Send > `Python verification_certificate.py <path_to_your_rootCA_cer> (i.e rootCA_certificate.cer) <path_to_your_rootCA_key> (i.e rootCA_key.pem) <aws_registration_code>`

Figure 70. Execute verification_certificate.py script

5. Check that the verification certificate is successfully generated in the **Provisioning** folder in your file system as shown in Figure 71:

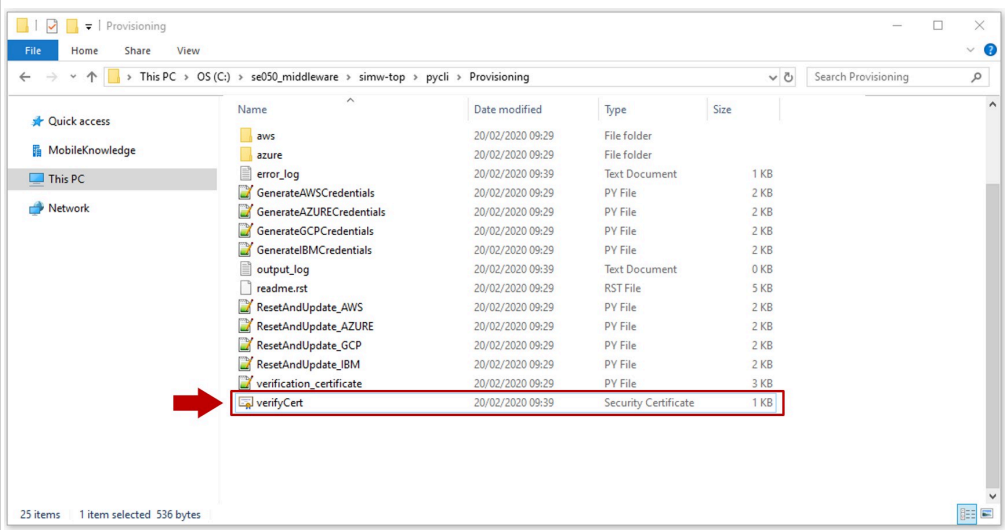

Figure 71. Verify generation of the AWS verification certificate

4.2.3 Upload root CA and AWS verification certificate

The registration of the root CA is completed after uploading it together with the AWS verification certificate. To register your root CA, follow these steps indicated in Figure 72:

1. Select from your file system the root CA certificate in .cer format (1)
2. Select from your file system the verification certificate in .cer format (2)
3. Check the boxes *Activate CA certificate*and *Enable auto-registration of device certificates* (3)
4. Click ***Register CA Certificate*** (4)

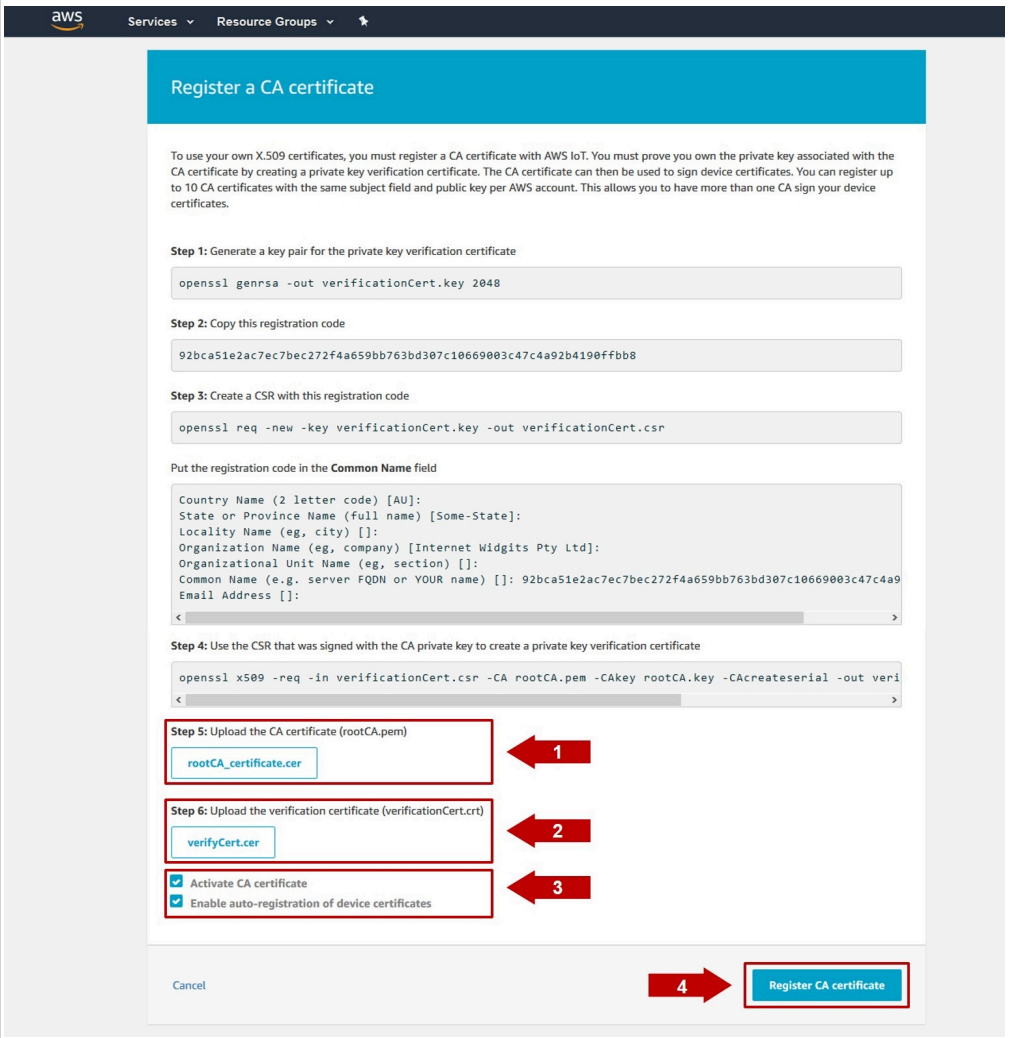

Figure 72. Upload your root CA and AWS verification certificate

5. When your root CA is registered successfully, it should now be visible in AWS dashboard and appear as ***Active*** as shown in Figure 73:

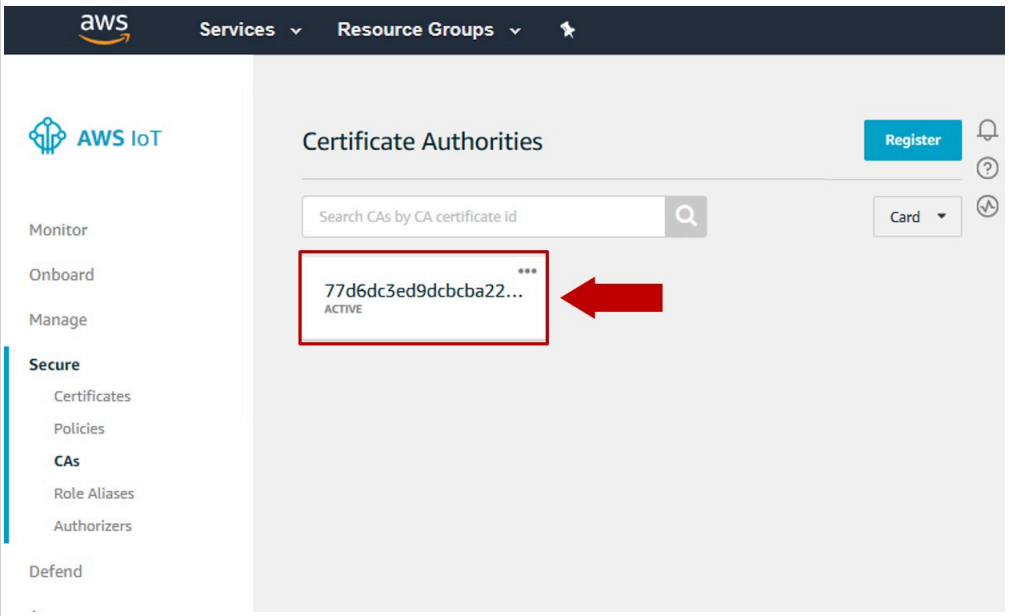

Figure 73. Check that your root CA is registered

4.3 Register device certificate

AWS IoT Core uses client certificates for device authentication. Any device that does not have a valid certificate signed by the registered root CA is denied access and cannot communicate with AWS IoT Core servers. To register client certificates to AWS IoT core, follow these steps:

1. From the AWS IoT Core dashboard, go to **Secure**, go to **Certificates** and click on the **Create a certificate** button as shown in Figure 74:

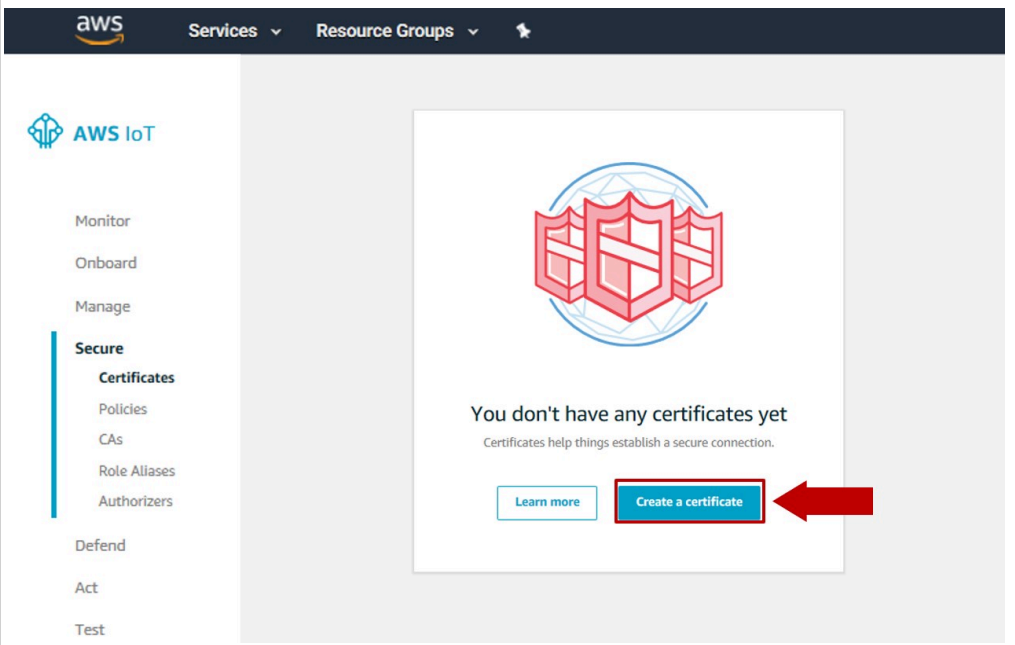

Figure 74. Go to the AWS IoT Certificates menu

2. A new menu called *Create a certificate* will be opened. Click on the **Get started** option as shown in Figure 75:

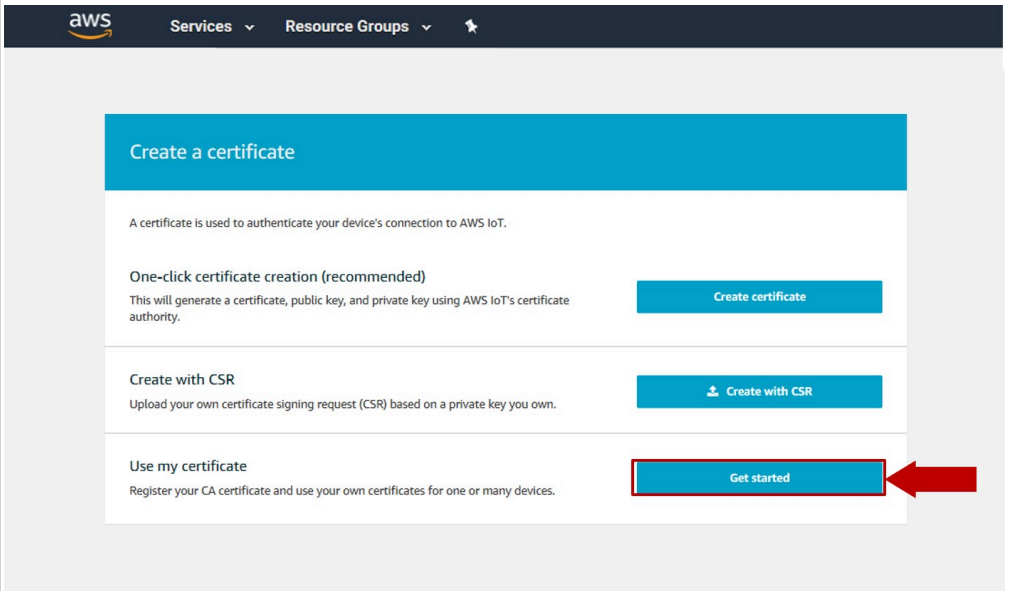

Figure 75. Create a certificate

3. A new menu called *Select a CA* will be opened. In this menu, the root CA you registered in Section 4.2 should appear. Select your root CA certificate and click on **Register certificates** button as shown in Figure 75

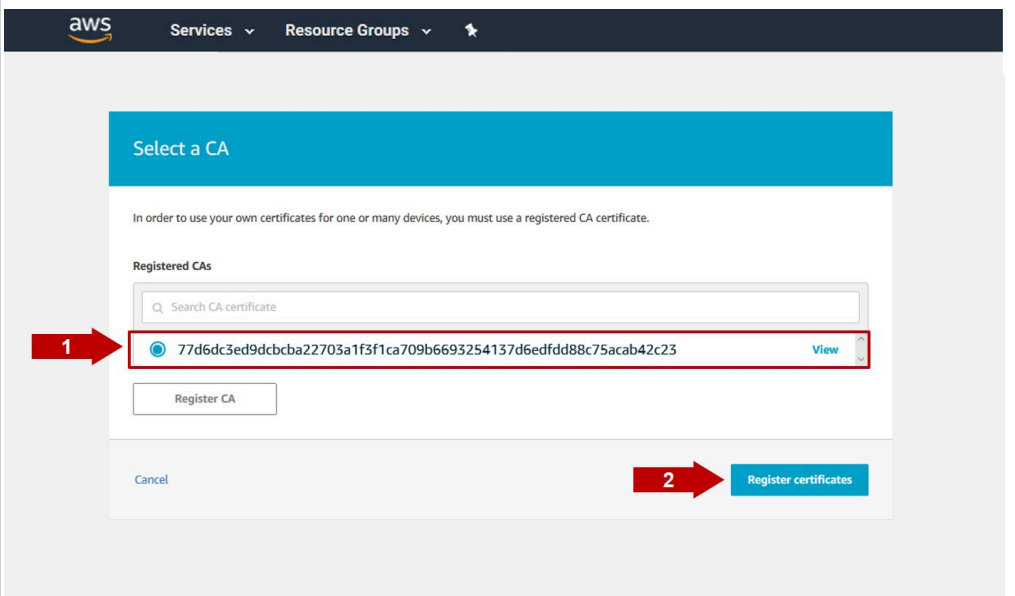

Figure 76. Select a CA to register a client certificate

4. Click on the button **Select certificates** (1), select from your file system the device certificate to be uploaded (2), and click **Open** button as shown in Figure 77 (3). This device certificate was generated in Section 4.1.3.

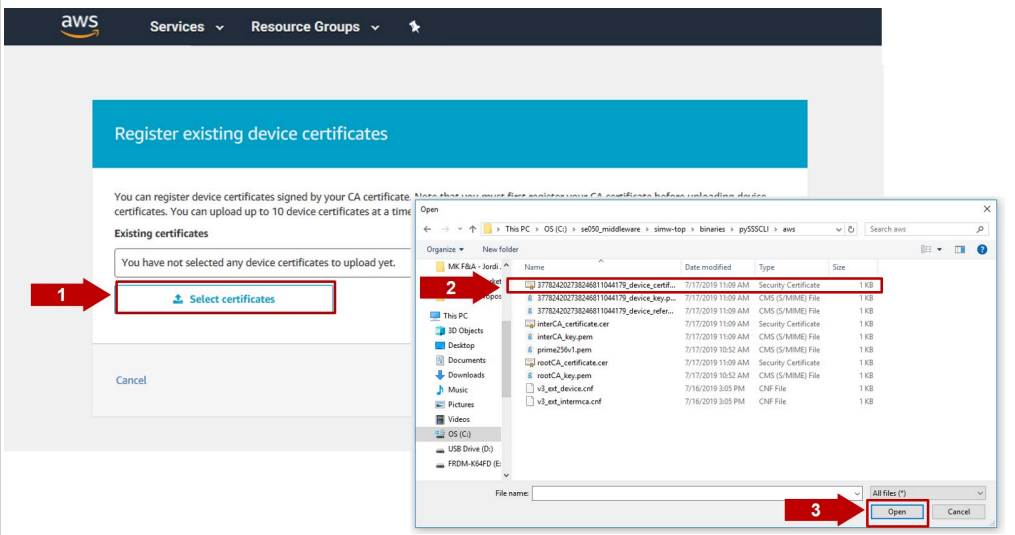

Figure 77. Select a client certificate

5. Select the option ***Activate all*** (1) and click on the button **Register certificates** (2) as shown in Figure 78.

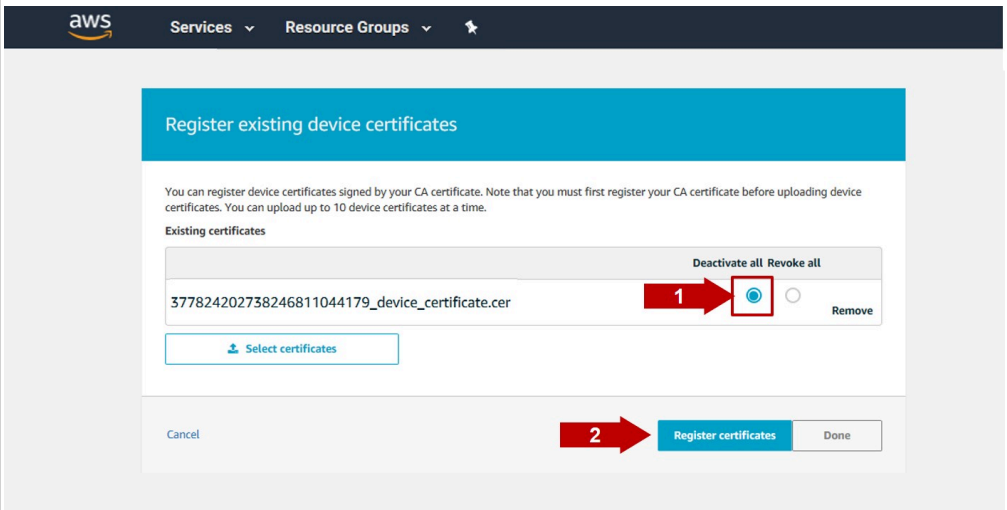

Figure 78. Register and activate device certificate

6. Your device certificate is now registered and visible in your AWS IoT Core dashboard as shown in Figure 79.

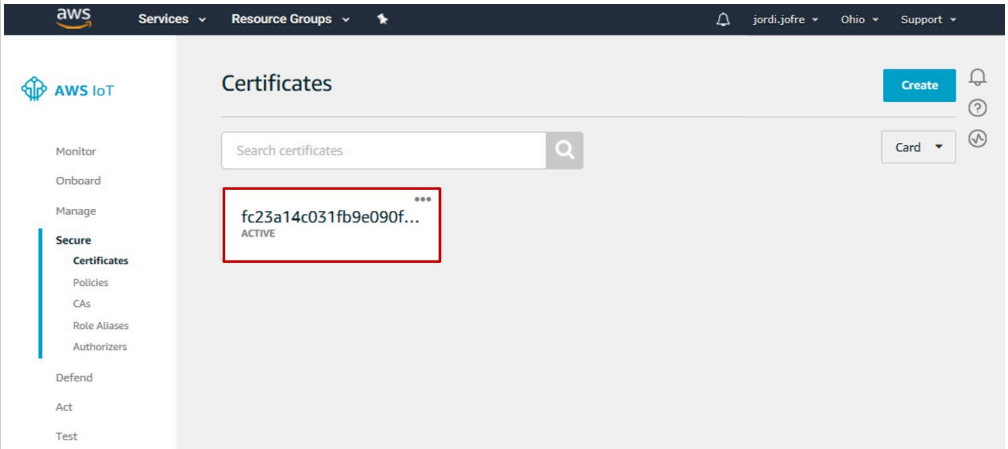

Figure 79. Confirm device certificate registration

4.4 Attach thing and policy to certificate

AWS IoT Core uses client certificates for device authentication. Any device that does not have a valid certificate signed by the registered root CA is denied access and cannot communicate with AWS IoT Core servers. To register client certificates to AWS IoT core, follow these steps:

1. Attach a thing to your certificate following the instructions shown in Figure 80.
   a. Click on the top right corner to go to the device certificate options.
   b. Click on **Attach a thing**.
   c. Select the AWS IoT Thing you created in Section 3.3. In this example, it was called **my_thing**.
   d. Click on the **Attach** button.

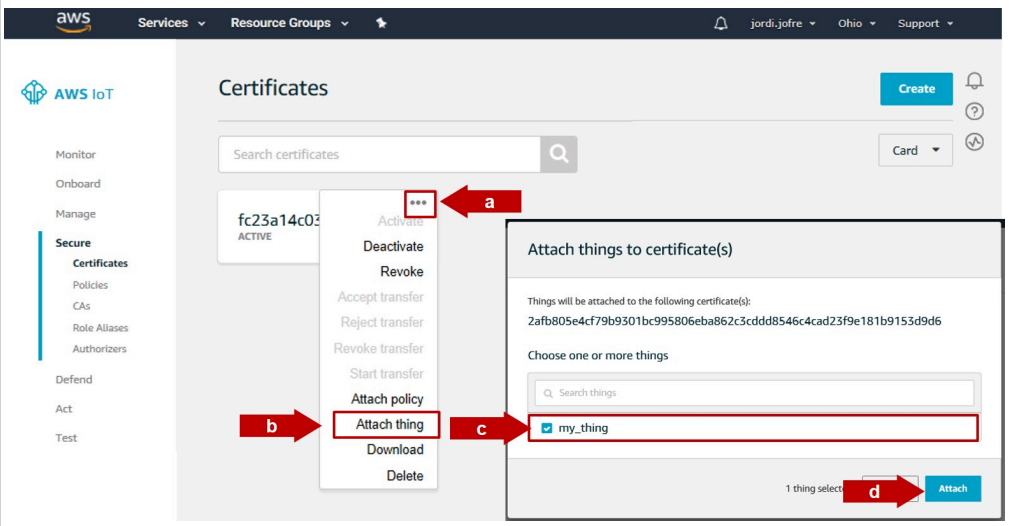

Figure 80. Attach a thing to your device certificate

2. Attach a policy to your certificate
   a. Click on the top right corner to go to the device certificate options.
   b. Click on **Attach a policy**.
   c. Select the AWS IoT Policy created in Section 3.4. In this example, it was called **my_policy**.
   d. Click on the **Attach** button.

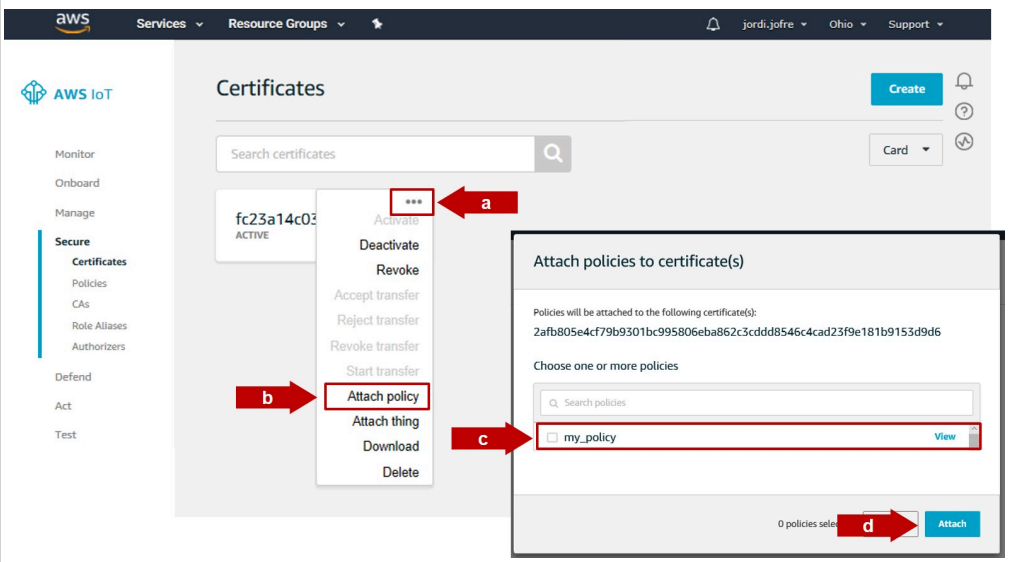

Figure 81. Attach a policy to your device certificate

4.5 AWS IoT Core project configuration

To run the AWS project example using the FRDM-K64F board, we need to:

- Download and install the FRDM-K64F SDK.
- Import AWS IoT Core example project
- Configure AWS IoT Core project account settings
- Execute AWS IoT Core example project

Note: *Before running the AWS IoT Core demo example, you need to have installed MCUXpresso IDE and FRDM-K64F SDK in your local environment and imported the AWS IoT Core 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 AWS IoT Core project example.*

4.5.1 Download and install the FRDM-K64F SDK

The AWS IoT Core device onboarding project example is included as part of the FRDM-K64F SDK. Install it to your MCUXpresso workspace as shown in Figure 82:

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.

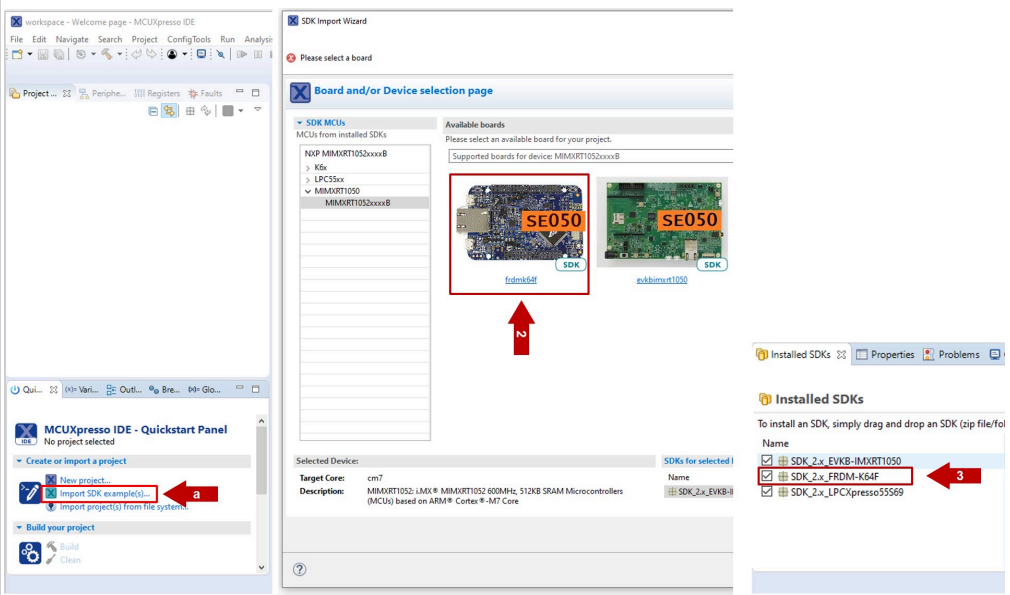

Figure 82. Import FRDM-K64F SDK

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.

4.5.2 Import AWS IoT Core example project

The FRDM-K64F SDK includes a project example called `se_SE050x_cloud_aws`. Import it to your MCUXpresso workspace as shown in Figure 83:

1. Click *Import SDK examples* from the MCUXpresso IDE quick start panel.
2. Select `se_SE050x_cloud_aws` 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

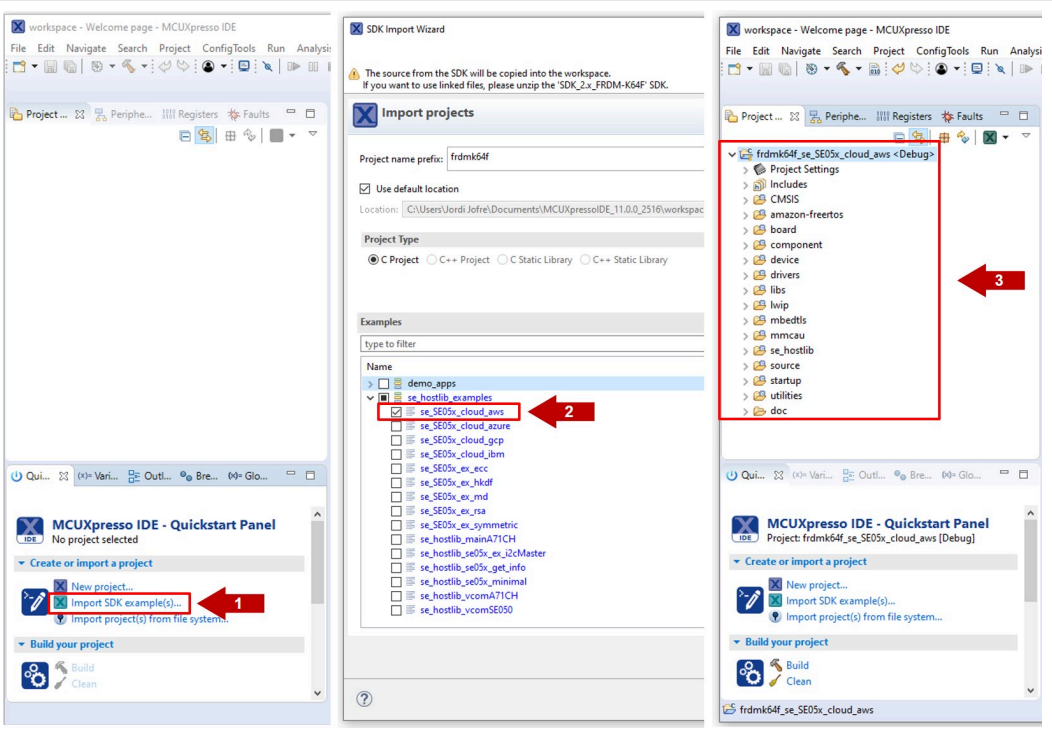

Figure 83. Import AWS project in the workspace

4.5.3 Configure AWS IoT Core project account settings

We need to change the AWS Rest API Endpoint in the MCUXpresso demo project with the one in your AWS IoT Core account settings. Follow these steps:

1. From the AWS IoT Core dashboard, go to *Manage*, then go to *Things* and click on your AWS IoT Thing as shown in Figure 84:

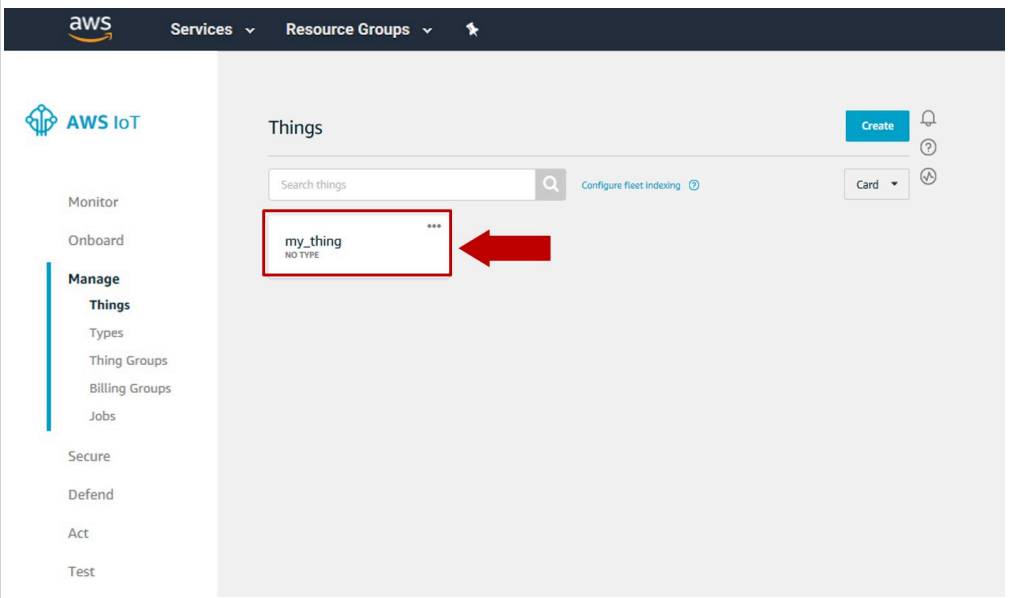

Figure 84. Go to your AWS IoT Thing

2. On the left hand side menu, (1) go to **Interact**. Inside this menu, you will find your (2) Rest API Endpoint as indicated in Figure 85. Copy this URL.

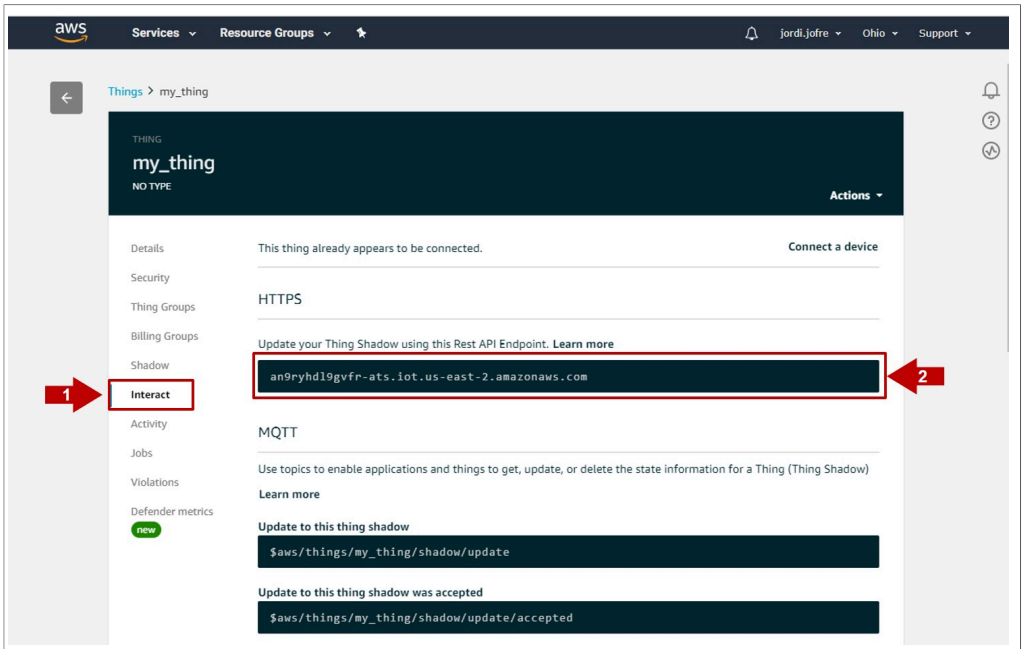

Figure 85. Find your Rest API Endpoint

3. Go to the AWS demo in your MCUXpresso workspace. Navigate to the `aws_clientcredential.h` file located in `frdmk64f_se_SE05x_cloud_aws\source` folder. Replace the `clientcredentialMQTT_BROKER_ENDPOINT`

variable with the Rest API Endpoint of your AWS account obtained in the previous step, as well as you thing name as created in Section 3.3. Check Figure 86 for reference.

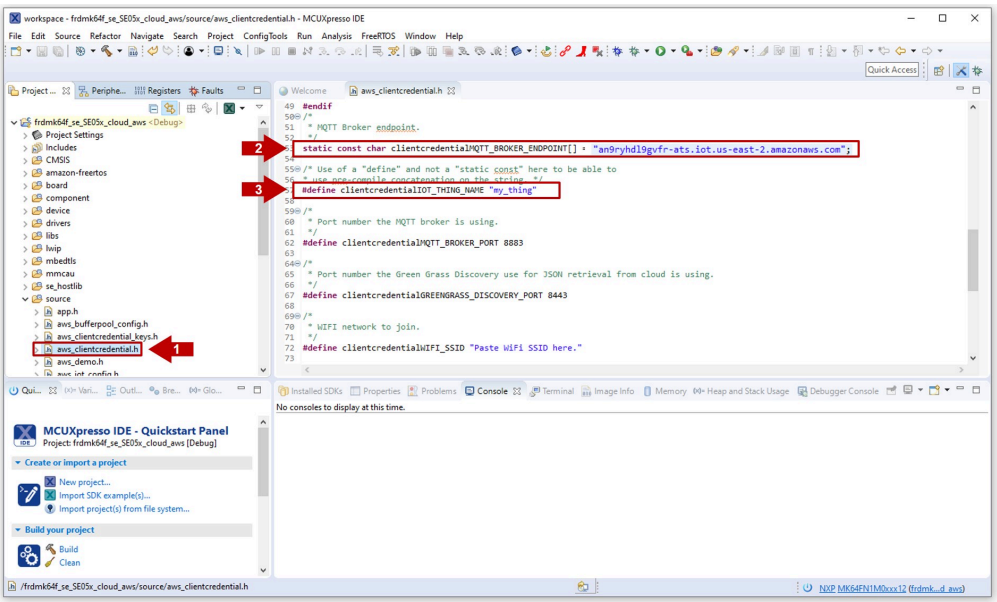

Figure 86. Set the Rest API Endpoint and thing name in your project settings

4. On the same **Interact** menu, you will find MQTT topics that enable applications and things to get, update, or delete the state information for an AWS thing. For instance, copy the MQTT ***update*** topic as shown in Figure 87:

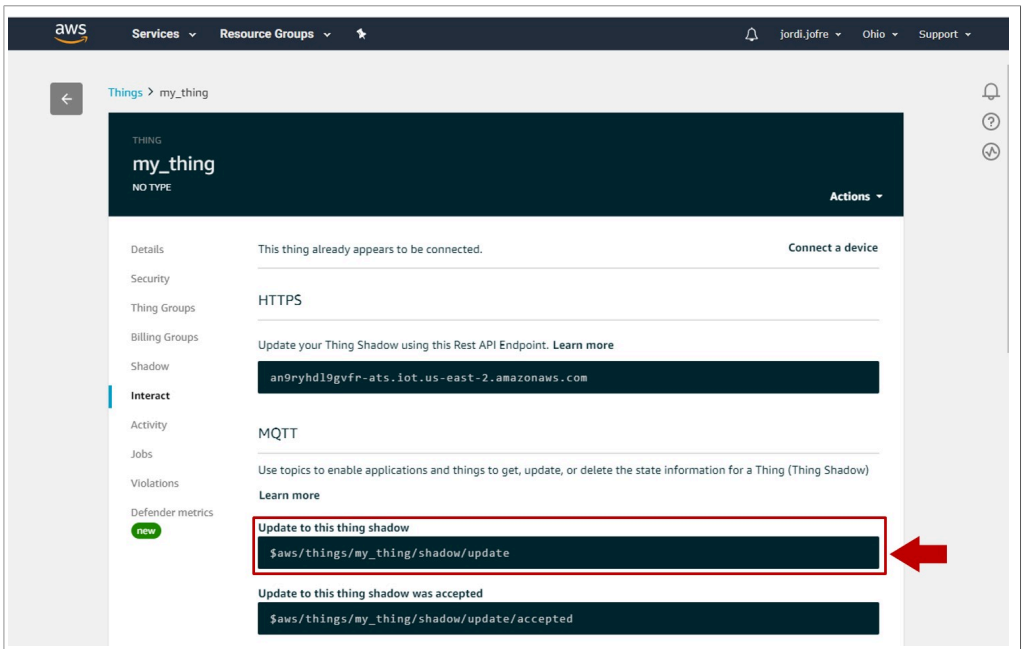

Figure 87. Find your MQTT *Update* topic

5. Go to the AWS demo in your MCUXpresso workspace. Navigate to the `aws_jitr_task_lwip.c` file located in `frdmk64f_se_SE05x_cloud_aws\source` folder. Replace the `#define PUB_TOPIC` variable with the MQTT topic you obtained in Figure 86 as shown in Figure 87.

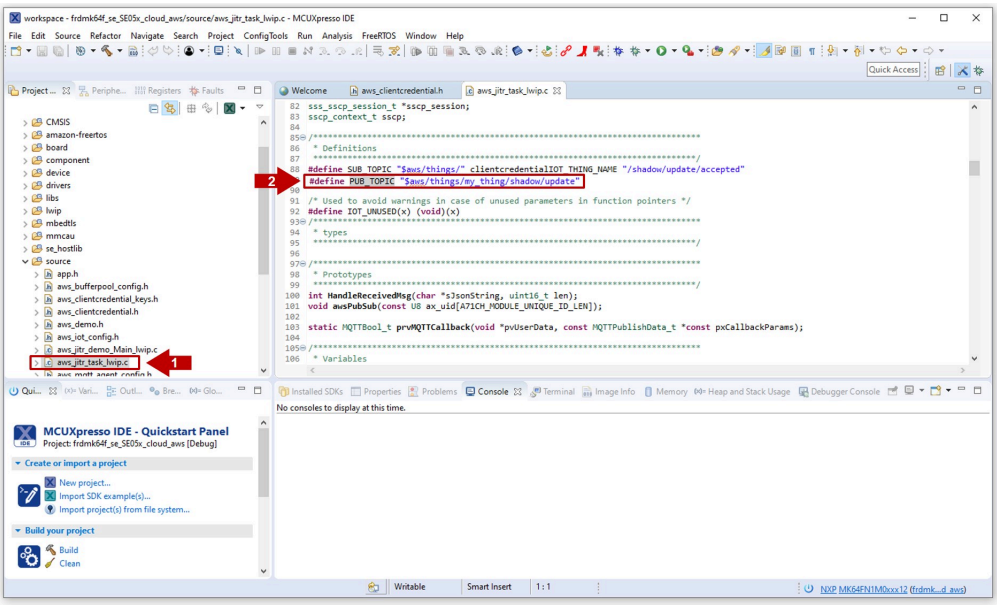

Figure 88. Set the MQTT publish topic

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.

NXP — wordmark and logo are trademarks of NXP B.V.

Tables

Figures

Contents

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 15 January 2021
Document identifier: AN12404
Document number: 535216