# MASTERs 2024 Lab Manual

***24042 SEC3***

**Enhancing Device Security:**

**Leveraging Trust MANAGER / keySTREAM solution to comply with security legislation and standards as TEC 31318 and AIS-140**A logo with a light bulb and a brain

Description automatically generated

**Lab Manual for *Enhancing Device Security:***

***Leveraging Trust MANAGER / keySTREAM solution to comply with security legislation and standards as TEC 31318 and AIS-140***

Table of Contents

[MASTERs 2024 Lab Manual 1](#_Toc170811856)

[Introduction: 3](#_Toc170811857)

[Prerequisites: 3](#_Toc170811858)

[Hardware Requirements: 3](#_Toc170811859)

[Software Requirements: 4](#_Toc170811860)

[Lab 1 – Create fleet profile and API keys 5](#_Toc170811861)

[Purpose: 5](#_Toc170811862)

[Overview: 5](#_Toc170811863)

[Procedure: 5](#_Toc170811864)

[Results: 15](#_Toc170811865)

[Summary: 15](#_Toc170811866)

[Lab 2 – Setting up KeySTREAM service and AWS. 16](#_Toc170811867)

[Purpose: 16](#_Toc170811868)

[Overview: 16](#_Toc170811869)

[Procedure: 16](#_Toc170811870)

[Results: 29](#_Toc170811871)

[Summary: 29](#_Toc170811872)

[Lab 3 – Setting up embedded project and observing output logs on Serial terminal and on AWS portal. 30](#_Toc170811873)

[Purpose: 30](#_Toc170811874)

[Overview: 30](#_Toc170811875)

[Procedure: 31](#_Toc170811876)

[Results: 45](#_Toc170811877)

[Summary: 45](#_Toc170811878)

[Appendix A - Appendix Title 46](#_Toc170811879)

[Appendix B - Appendix Title 47](#_Toc170811880)

## Introduction:

This lab demonstrates the pre-provisioned ECC608 TrustMANAGER device and the keySTREAM™ SaaS in-field provisioning and key management. The ECC608 will go through a take ownership process where the keySTREAM will set a custom chain of certificate from the SaaS and key the root and intermediate private keys protected in Kudelski cloud HSMs.​

**Upon completion, you will:**

* Use Kudelski KeySTREAM web portal to create a product line with Root keys (Fleet Management)
* Use TPDS to upload the device manifest and upload device root cert to AWS
* Observe the device connecting to the KeySTREAM service and use device certificate renewal and refurbish functions!

## Prerequisites:

The lab material assumes you have prior experience with:

* ECC608 device and Cryptography fundamentals (Asymmetric authentication, X509 certificates and TLS)
* MPLAB X IDE
* MPLAB based Programming/Debugging fundamentals
* C language programming

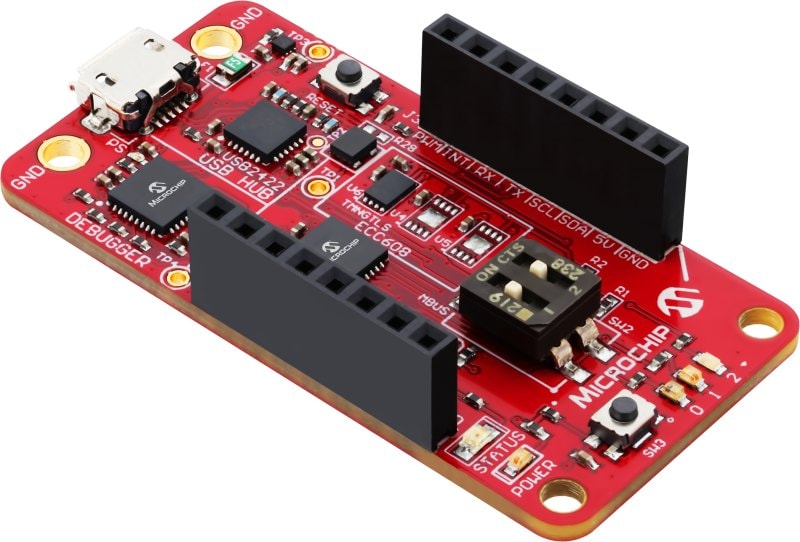
## Hardware Requirements:

Here is the list of hardware needed to complete the labs:

* [Trust Manager Development Kit](https://www.microchip.com/en-us/development-tool/EA06V72A)
  + EA06V72A - CryptoAuth Trust Manager Development Kit
  + 1x USB micro-B cables

A pair of black cables

Description automatically generated



* [WIFI 7 CLICK](https://www.mikroe.com/wifi-7-click)



## Software Requirements:

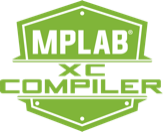
Here is the list of software needed to complete the labs:



[**Trust Platform Design Suite**](https://www.microchip.com/en-us/products/security/trust-platform/tpds)

****[**MPLAB® X IDE**](https://www.microchip.com/en-us/tools-resources/develop/mplab-x-ide)

**(version 6.20)**

****

[**MPLAB® XC32 Compiler**](https://www.microchip.com/en-us/tools-resources/develop/mplab-xc-compilers/xc32)

**A green and white hexagon with white text

Description automatically generated**

[**MPLAB® Code Configurator Plug In**](https://www.microchip.com/en-us/tools-resources/configure/mplab-code-configurator)

A computer with a letter on it

Description automatically generated

[**TeraTerm**](https://teratermproject.github.io/index-en.html)

# Lab 1 – Create fleet profile and API keys

## Purpose:

In this lab we will be creating a keySTREAM Fleet Profile (illustrating creation of a product line). As part of the creation of Fleet profile we will also be generating the Root CA certificate, with custom information about certificate names and expiry dates. Then we will create API keys to let TPDS make changes to the account.

## Overview:

A Fleet of chipsets is managed by means of a keySTREAM Fleet Profile. A Fleet Profile represents common characteristics of a set of chipsets from a keySTREAM perspective, that are required for proper operation. To properly onboard a chipset when in the field, the chipset must have an associated Fleet Profile Public Uid within the chipset, and a matching Fleet Profile within keySTREAM.

## Procedure:

We will be creating a fleet profile under KeySTREAM web portal. You can access the kudelski webportal [here](https://mc.obp.iot.kudelski.com/products/iot).

#### Step 1: Creating Kudelski keySTREAM account.

We will be creating a Kudelski account specific to a customer. We need this to configure and create Fleet profile which will then be used to run the lab demo.

##### 1.1: Go to the following link for account creation

Register link: <https://mc.obp.iot.kudelski.com/products/iot>

##### Login page screen shot1.2: Click on register

##### A screenshot of a login form Description automatically generatedA screenshot of a register Description automatically generated1.3: Provide information required for account creation, a verification email will be sent out the provided email.

##### A screenshot of a computer Description automatically generated1.4: Look out for the verification code on the registered email. Enter the verification code once it is available to you. Once the right code is entered you should see the successful account creation message.

A screenshot of a computer

Description automatically generated

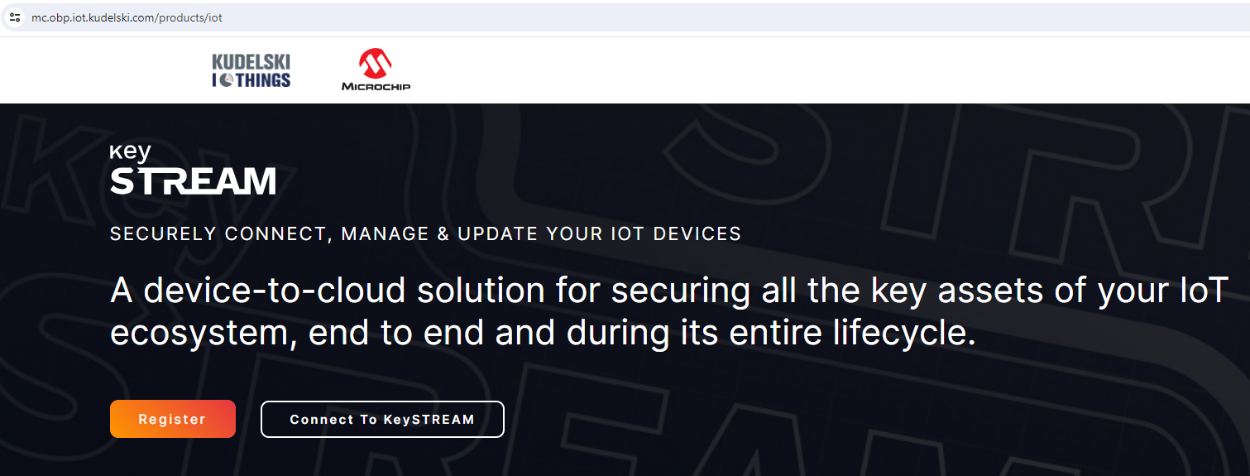
#### Step 2: Creating fleet profile under KeySTREAM portal

In this step we will create a fleet profile under the Kudelski keySTREAM portal. Fleet profile could represent a product line or a subdivision under a product line. For example, you can set up a fleet profile for a line of internet connected thermostat, this will let you remotely manage those specific devices.

During the creation of Fleet profile, we will provide information required for creating Root CA that will be used for these specific set of devices. All the devices under the fleet profile will be signed by this Root CA certificate. Each created fleet profile will always have a Root CA certificate associated with it.

##### 2.1: Login to the Kudelski keySTREAM portal

Login to the Kudelski keySTREAM portal using the [link](https://mc.obp.iot.kudelski.com/products/iot). Click on connect to keySTREAM to login.



Once logged in you will be redirected to the home page.

##### 2.2: Creating a new Fleet profile

(i) On the let navigation pan click on “Fleet & PKI”

A screenshot of a phone

Description automatically generated

(ii) Click on “Create” under “2-Tier Root CA”.

A screenshot of a computer

Description automatically generated

###### Step 2.2.1.1: Create your Root Certificate Authority

Now we will be adding information required for the Root CA Certificate creation for this Fleet profile.

A screenshot of a computer

Description automatically generated

Enter details regarding the Root CA certificate creation. You can use your organization name or any preferred string for the name fields.

Root CA Common Name: Information for the Common name field of x509 certificate. Accepts **user defined string** of up to 16 characters.

Root CA Organization: Information for the Root organization name field of x509 certificate. Accepts **user defined string** of up to 16 characters.

Root CA Certificate Validity (years): Number of years of validity of the root certificate authority. Enter **30 years** for the demo.

Once the root CA is created ,you will receive a pop-up as below. Click ‘Close’.

A white background with black text

Description automatically generated

You can find the Root CA created and listed below.

A screenshot of a computer

Description automatically generated

Step 2.2.1.2: Fill out the required fields in Fleet & Cert

1. Click ‘**Create**’ under ‘***Fleet & Cert***’ tab as below.

A screenshot of a computer

Description automatically generated

(ii) Fill in the required details.

A screenshot of a computer

Description automatically generated

***Fleet Profile Public Uid*** (required): The Fleet Profile Public Uid identifies this group of devices. This identifier will be required in your embedded system on the device side.

Manufacturer (optional): identifying the Manufacturer of the device.

Brand (optional): identifying the brand of the device.

Model (optional): identifying the model of the device.

Once all the required fields are filled , click on ‘**Next**’ to proceed.

A screenshot of a computer

Description automatically generated

Enter details regarding the 2-tier Intermediate CA certificate creation. You can use your intermediate CA organization name or any preferred string for the name fields.

2-Tier Intermediate CA Common Name: Information for the Common name field of x509 certificate. Accepts **user defined string** of up to 16 characters.

2-Tier Intermediate CA Organization: Information for the Intermediate CA organization name field of x509 certificate. Accepts **user defined string** of up to 16 characters.

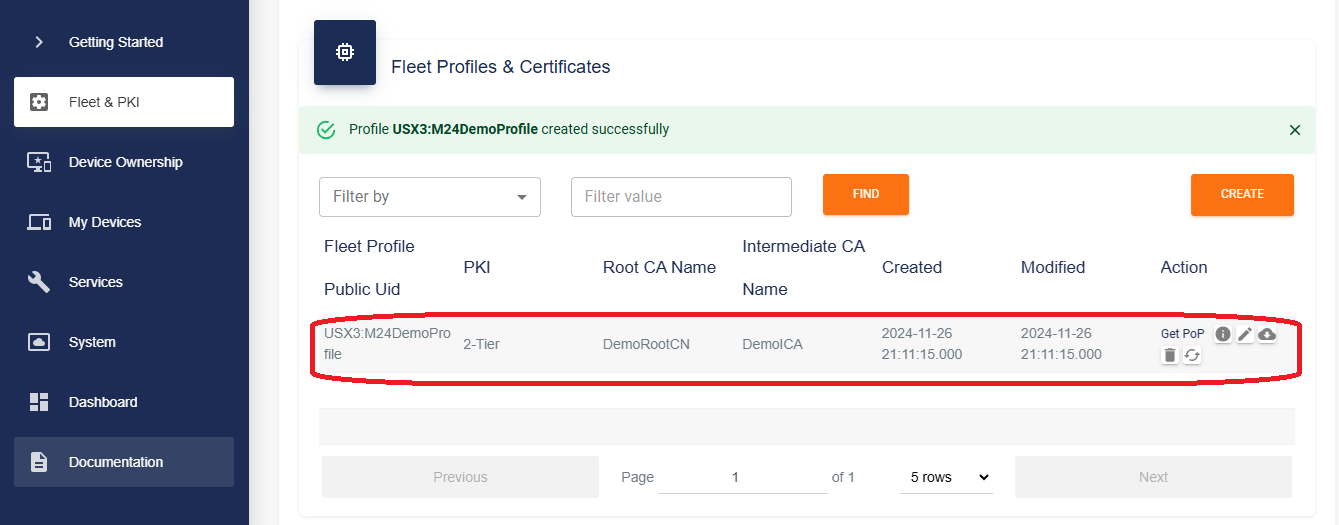
2-Tier Intermediate CA Certificate Validity (years): Number of years of validity of the root certificate authority. Enter **25 years** for the demo.

Device Operational Certificate Validity (in years): The number of years of validity of the Device Operational Certificate. Enter **10 years** for the demo.

Automatic Leaf Certificates Renewal (Optional): This field is optional. When enabled, keySTREAM will automatically renew the Device Operational Certificate the defined number of days before it expires. This should be **Disabled** for the demo.

Once all required fields are filled click on the **Commit** button.

Verify your new Fleet Profile is listed Under Fleet profiles and certificates.



That’s it, you have successfully created a fleet profile for your product, during that process you also created the Root CA associated with the Fleet Profile.

**Step 3: Creating API keys.**

API key is used to authenticate a user/program to a keySTREAM API. This is useful for managing devices through user written scripts and routines once the device is in production. For this lab we will create API keys and provide them to TPDS so it can connect to your account and help set up your account. You will learn more about what TPDS does in the next lab section.

Step 3.1: Navigate to “System” section in the left navigation pane.

A screenshot of a phone

Description automatically generated

##### Step 3.2: Creating API keys.

With the System window open follow the steps below to create the API keys

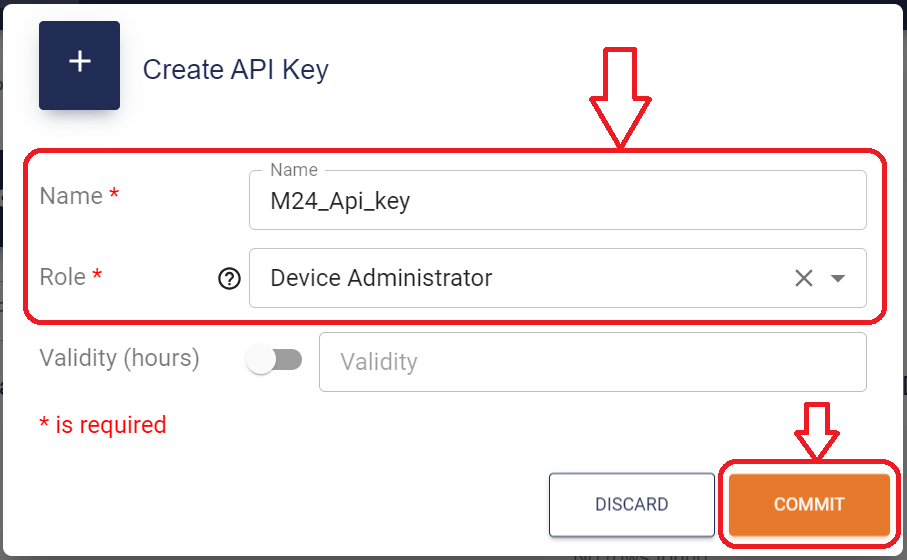
###### Step 3.2.1: Click create.

Step 3.2.2: Enter details regarding the API key.

Name (required): Any user defined name, identifying the API key. For the lab let us can use “**M24\_Api\_key**”

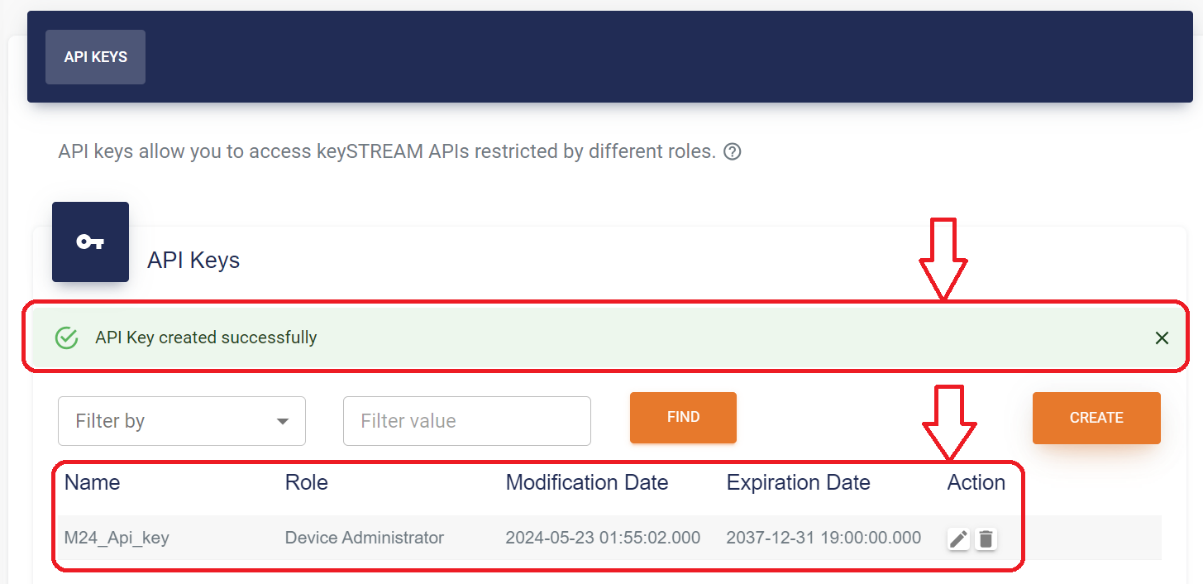
Role (required): Defines access restrictions and permissions. TPDS requires administrator access to the account so let’s select “**Device Administrator.**”

Validity (optional): The number of hours for which the API key must be valid, from its creation. This field is typically used when you want to give a person/program temporary access to your account. For the lab, **do not enable, leave it empty**.

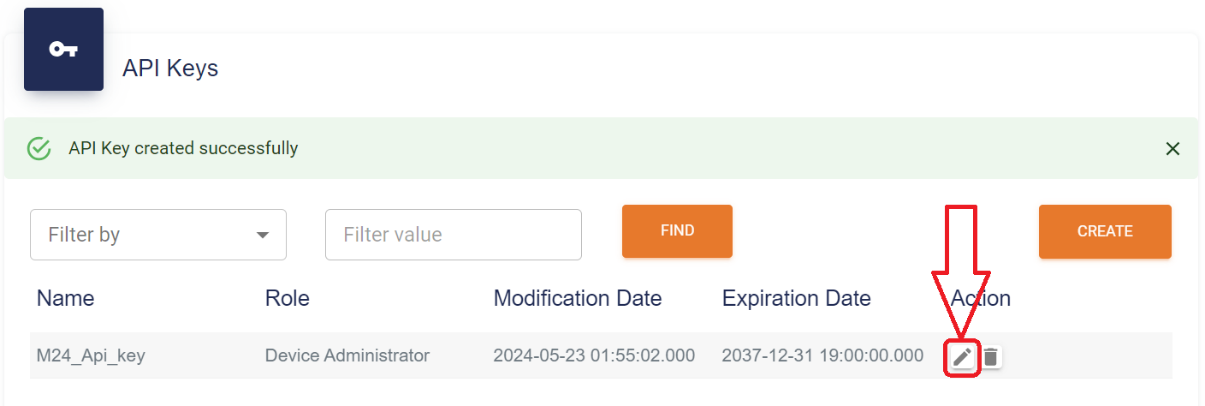


Once the details are filled in, click on “**Commit**” button.

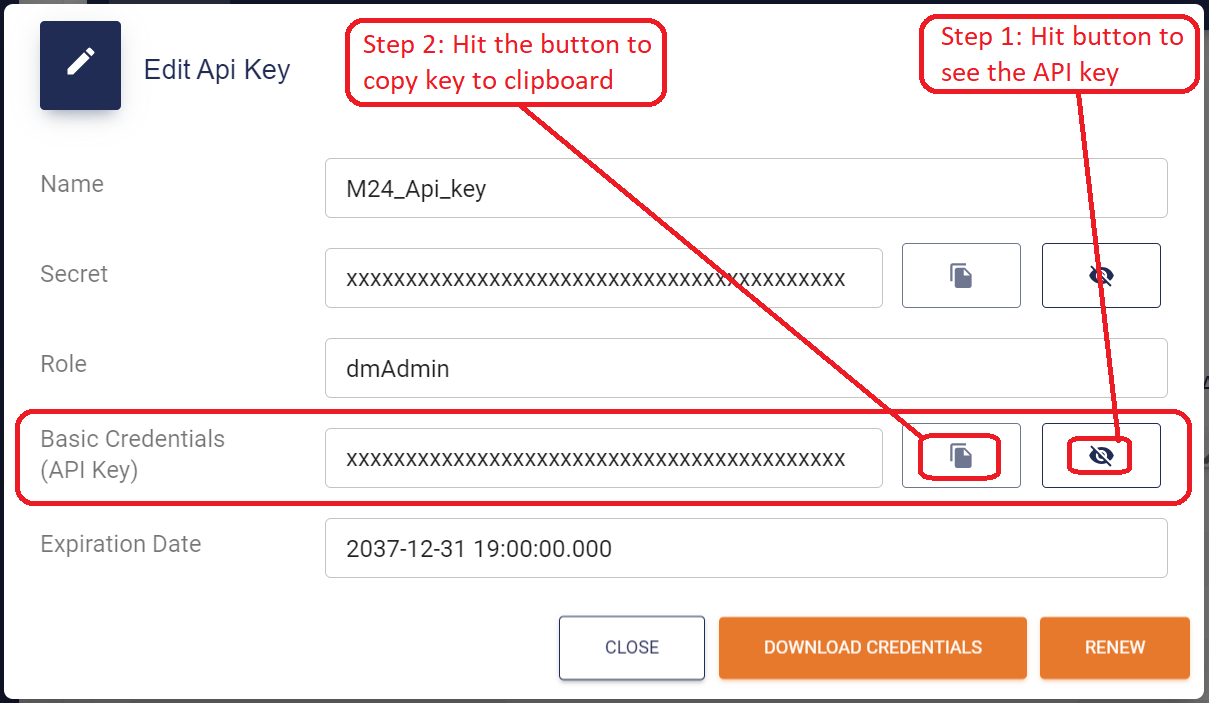
Step 3.2.3: Getting access to the generated key  
Once the key is generated , you will see the success message. Verify the key you created, is populated in the table below.



Now let us look at the key we just generated.  
  
 Click on the edit button under the Actions column.



This will open up “Edit API key window.”



We will need this key for the future steps, keep a copy of the key. You can always go to portal again and copy the key when required.

That is it, you have created the API key and learned how to get access to the generated key.

## Results:

After these steps you should have a KeySTREAM account created for you along with Fleet profile UID representing a product you would like to manage remotely.

## Summary:

1. We created an account on KeySTREAM.
2. Created a Fleet profile, this would represent a product line you will manage remotely.
3. Created an API key, we will be using this key to get access to the KeySTREAM account through APIs.
4. We are also slowly getting used to navigating KeySTREAM UI.

# Lab 2 – Setting up KeySTREAM service and AWS.

## Purpose:

We will be claiming the device into the Kudelski KeySTREAM account and uploading root certificates of the device profile to AWS.

## Overview:

Two main steps are done in this lab.

Device claiming – The process of getting details of the hardware you have on your desk and uploading the details(manifest) so you can claim the device into your account.

Uploading Root CA cert to AWS – We will then be uploading the Root CA certificate we created in Lab 1 to AWS account through TPDS. This is to ensure AWS can validate the devices trying to connect to your account.

## Procedure:

We will go through steps to claim the device we have on KeySTREAM service and then go through the process of uploading Root CA certificate of the fleet profile to AWS. All these steps will be done through Trust Platform Design Suite (TPDS).

**Terminology guide:**

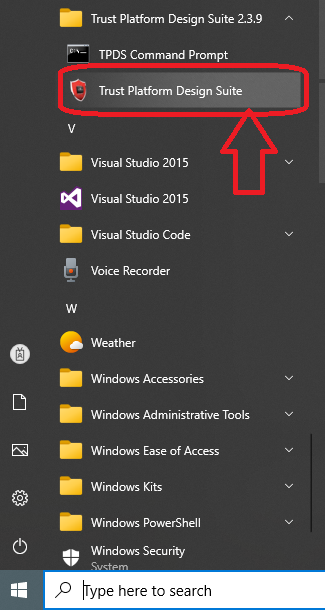
|  |  |
| --- | --- |
| TPDS | Trust Platform Design Suite, a Microchip software that helps with evaluation of crypto products |
| Manifest | Manifest files contain information about a single (or) multiple device. This provides a way to link a Microchip Trust Security Device for a given customer to the Kudelski keySTREAM infrastructure. |
| Root CA Certificate | A root certificate is a type of digital certificate that is self-signed and used to verify the identity of the root certificate authority (Root CA) in a chain of trust. |
| AWS | AWS – Amazon Web services |

#### Step 1: Open KeySTREAM use case under Trust Platform Design Suite (TPDS)

Steps to help navigate to the Kudelski KeySTREAM use case under TPDS.

##### 1.1: Open TPDS

Click on Start and Navigate to Trust Platform Design Suite 2.x.x menu and click on Trust Platform Design Suite.

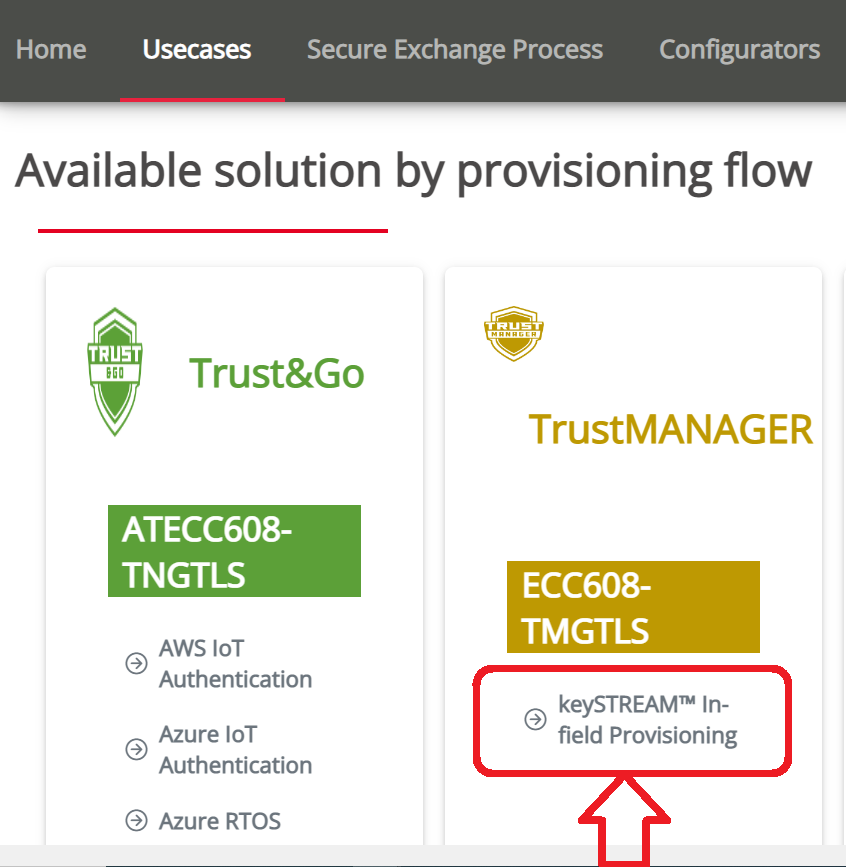
****

##### 1.2: Go to Usecases section

##### 

*1.3: Open “KeySTREAM infield provisioning” usecase*

Once under the Usecases section scroll down and navigate to the “*Available solutions by provisioning flow*” section. You will see the “KeySTREAM infield provisioning” use case listed under TrustManager column. Click on **KeySTREAM infield provisioning** to open the use case.

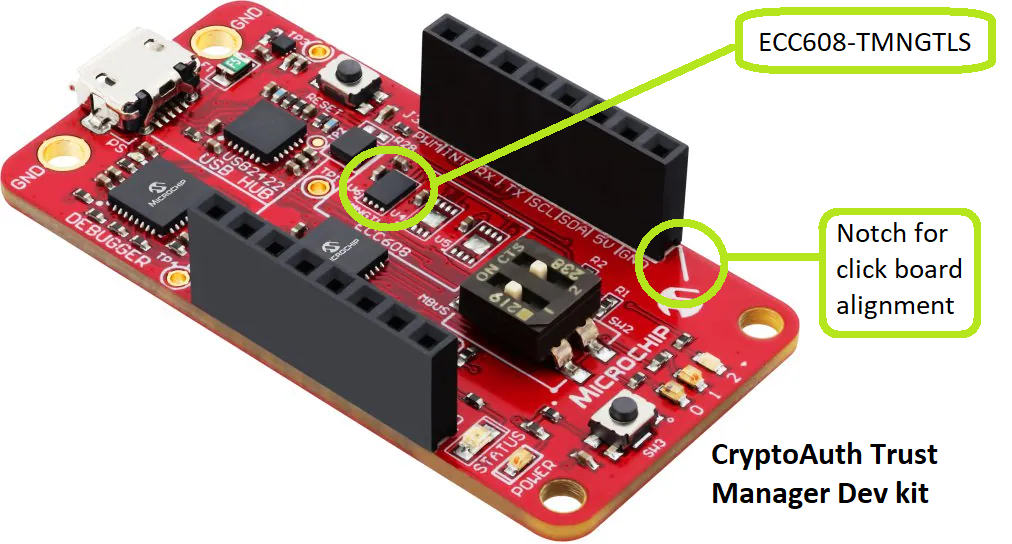
  
  
1.4: Verify the use case is open  
Once the use case notebook loads it should look like the image below.  
  
A screenshot of a document

Description automatically generated

##### With this we are done with Step 1, we have successfully opened TPDS and navigated to the KeySTREAM Infield provisioning use case notebook.

##### **Step 2: Generating/uploading device manifest to Kudelski KeySTREAM account.**

We will be generating the manifest for the ECC608-TMNGTLS device soldered on the CryptoAuth Trust Manager Development Kit. Once we have the device information in a manifest you will be able to claim the device into your account. More information on claiming the device will be explained in the steps below.

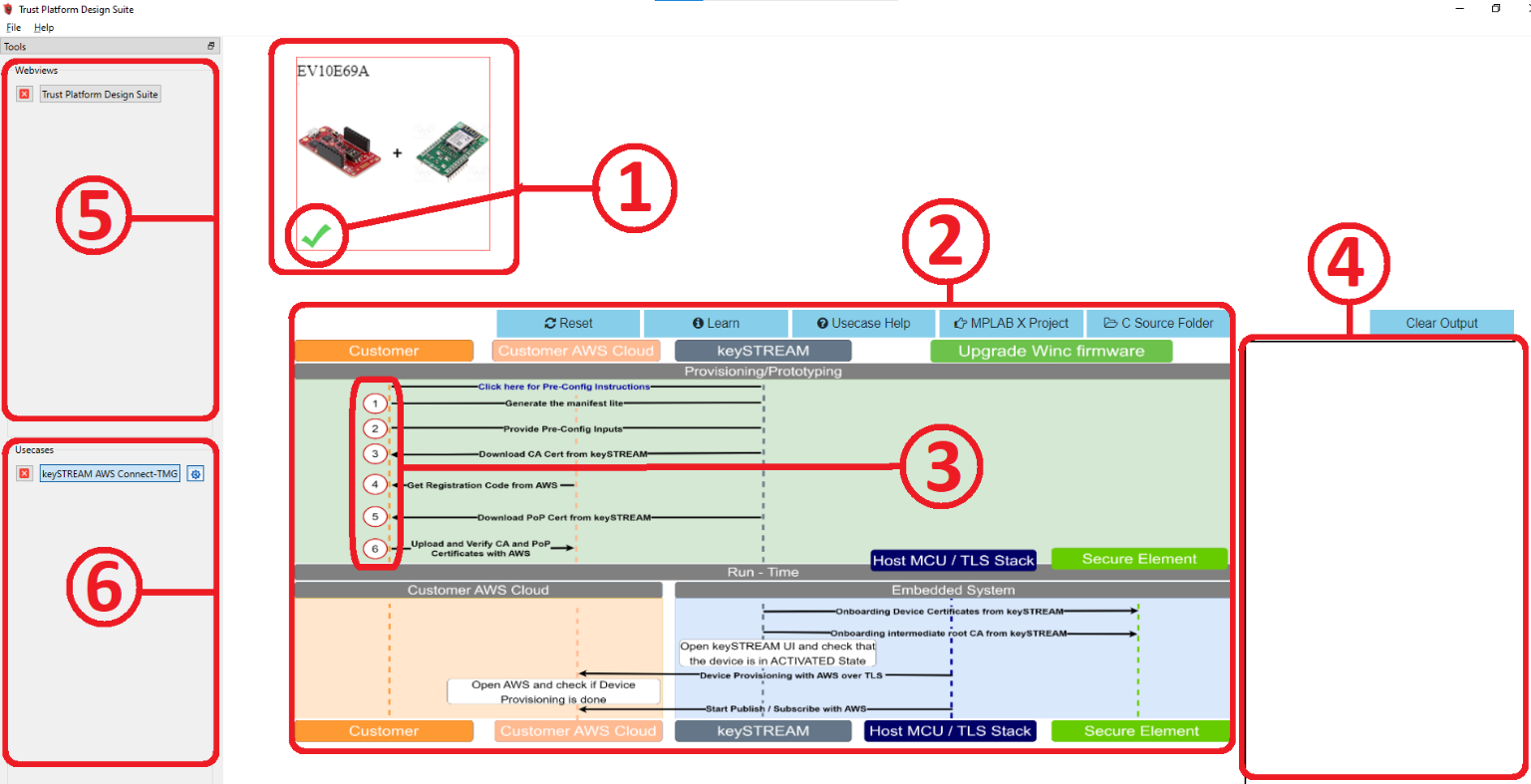


##### 2.1: Hardware connections

1. Connect the wifi 7 click board to the CryptoAuthTrust Manager dev kit, make sure the notch on the wifi 7 click board aligns with the notch indication on the CryptoAuth Trustmanager dev kit.
2. Make sure the boards are perfectly aligned; no pins should be visible after the connection. Misaligned pins can lead to permanent damage to the dev kit.
3. Connect the CryptoAuth Trust Manager dev kit to PC using the provided micro-USB cable.

##### 2.2: Brief intro – TPDS (Trust Platform Design Suite) Usecase

For people new to TPDS software and the use cases here is an image explaining most aspects of use case page which we will use in the lab session.



|  |  |  |
| --- | --- | --- |
| 1 | Board selection window | Ensure the board is selected. Green tick indicates a successful selection. |
| 2 | Usecase diagram window | Contains usecase execution buttons with more utility buttons on top (the blue buttons) |
| 3 | Usecase execution buttons | These are clickable buttons used to run the usecase steps. |
| 4 | Output window | When usecase steps are executed, the output will be displayed here |
| 5 | Webview pane | All webviews and help pages are listed here. |
| 6 | Usecases pane | All usecases currently open are listed here |

##### 2.3: Generating manifest and claiming it into keySTREAM account

A blue circle with a black letter in it

Description automatically generated**What is a manifest ?** Manifest is a file that describes a CryptoAuth device, this is a custom json file defined by Microchip. Read more[here](https://ww1.microchip.com/downloads/aemDocuments/documents/SCBU/ProductDocuments/ReferenceManuals/Trust-Platform-Manifest-File-Format-60001759.pdf)

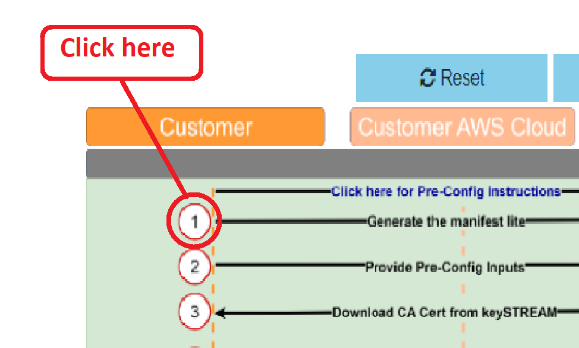
In our lab we will be generating manifest for the [ECC608-TMNGTLS](https://www.microchip.com/en-us/product/ECC608-TMNGTLS) device. This will have information like serial numbers and some public data. This file is then uploaded to keySTREAM portal to claim that specific device into your account.

Step 2.3.1: Generate manifest.

A blue circle with a black letter in it

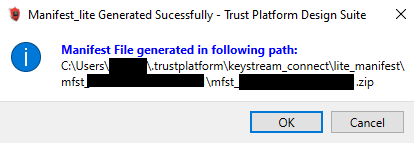
Description automatically generated**INFO** CryptoAuth TrustManager dev kit needs to be connected before clicking on step 1.

In the usecase diagram click on step 1



Once the execution is complete, you will get a message box pop up explaining where the manifest got generated. Keep a note of where this is generated. You will need the location in the future steps. It usually follows the following structure.

|  |
| --- |
| C:\Users\xxxx\.trustplatform\keySTREAM\_connect\lite\_manifest\mfst\_xxxxx\mfst\_xxxxx.zip |

**

The output window will also list the output of the execution.

A close-up of a message

Description automatically generated

Step 2.3.1: Claiming manifest under keySTREAM portal.

Now that we have a manifest, we are ready to claim it into the keySTREAM account.

Open keySTREAM web portal and navigate to “***device ownership***”.

A screenshot of a phone

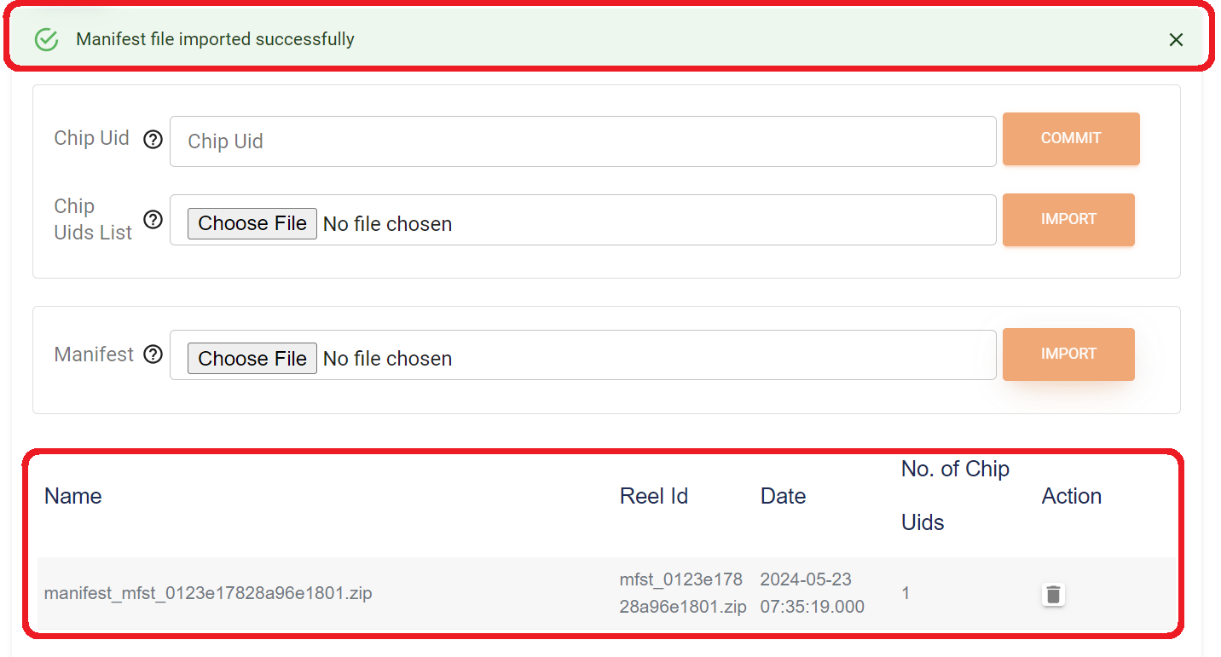
Description automatically generated

Under the device ownership tab, you will see different options through which you can claim the device. We will be claiming the device by using Manifest. Click on the **Choose file** button and navigate to where we generated the manifest. Typically, the manifest will be available under the following location.

|  |
| --- |
| C:\Users\xxxx\.trustplatform\keySTREAM\_connect\lite\_manifest\mfst\_xxxxx\mfst\_xxxxx.zip |



After choosing the manifest file, click on the import button. The devices in the manifest file will be claimed and you should see output similar to the image below.



A blue circle with a black letter in it

Description automatically generated**INFO** Ensure your device manifest is listed under Device Ownership -> Device Claiming

With this step you have generated a manifest of the device on your dev kit and claimed it into your account.

##### 2.2: Priming TPDS with all required data

In this step we will run step 2 in TPDS keySTREAM usecase. This step takes in the following inputs.

A screenshot of a computer

Description automatically generated

Fleet Profile Public UID: We will give the name of the fleet we created. Refer to page 15.

Wifi SSID: Enter Wifi SSID, refer to your help sheet.

Wifi Password: Enter Wifi SSID, refer to your help sheet.

keySTREAM Auth Token: Enter the API key we generated here.

AWS Access Key ID: Enter AWS Access Key ID, refer to your help sheet.

AWS Secret Access Key: Enter AWS Secret Access Key, refer to your help sheet.

Region: Enter AWS account region, refer to your help sheet.

Once all the information is filled in, click the **Submit** button. That is, now TPDS usecase has all the information that it needs to connect to keySTREAM service and run the rest of the lab session.

##### 2.3: Uploading the Root CA certificate associated with fleet profile to AWS

AWS has a defined process through which you can upload a root CA cert.

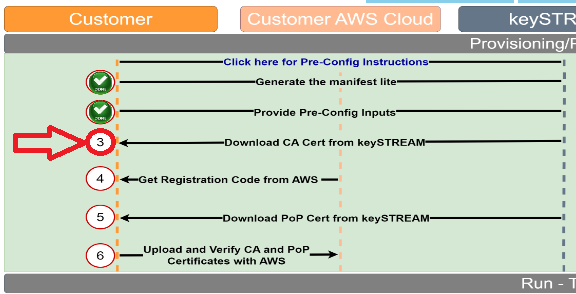
AWS process for uploading a Root CA certificate for AWS IOT core:

* Download Root CA cert.
* Get a registration code from AWS for uploading Root CA certificate.
* Create a CSR with details of Root CA certificate with AWS registration code under the Common name field.
* Use the Root CA key to sign the CSR, now we have what is called a POP (Proof of possession) certificate.
* Provide Root CA certificate and the POP certificate to AWS.
* AWS will validate the POP certificate with AWS registration code using the Root CA certificate’s public key. Once it passes validation, the root CA certificate is added to your AWS account.

TPDS (step 3 through step 6) will make it easier for users to do the above operations. Now let us look at each step in detail.

Step 2.3.1: Downloading the CA certificate associated with Fleet Profile UID

Click on Step 3



In this step, TPDS will use the keySTREAM credentials (API token) we provided on step 2.2 along with the Fleet profile UID to download the CA cert. This CA certificate is specific to the Fleet profile UID, hence the Fleet profile UID requirement.

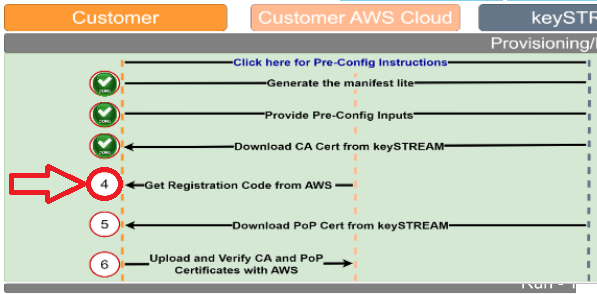
After TPDS is done executing the step, the output window should have an output like the image below. TPDS downloads and prints the CA certificate associated with the provided Fleet Profile UID. The certificate printed is encoded with base64 encoding following ASN.1 speciation, this can be decoded with an ASN.1 decoder.

A screenshot of a computer screen

Description automatically generated

Step 2.3.2: Getting registration code from AWS.

Click on Step 4



In this step, TPDS will use the AWS credentials (Access Key, Secret Access key and region) we provided on step 2.2 to get a registration code from AWS. This is a necessary step as AWS requires new CA certificate additions to be validated. This registration code will be used to create a CSR where the common name is appended with the registration code and signed by the root. This will be done in the next step.

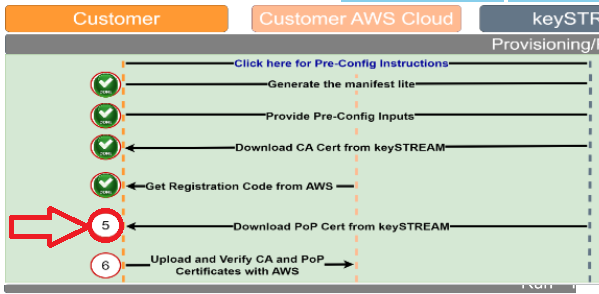
Once TPDS downloads the registration code from AWS, it prints it on the output window. The output should look something like this.

A computer screen shot of a code

Description automatically generated

Step 2.3.3: Downloading POP certificate from KeySTREAM.

Click on Step 5



Now that TPDS has the registration code from AWS and CA certificate from keySTREAM, it creates a CSR using root information with AWS registration code appended to the CSR common name. All this information is fed back to keySTEAM, the signed certificate (POP certificate) is downloaded from keySTREAM.

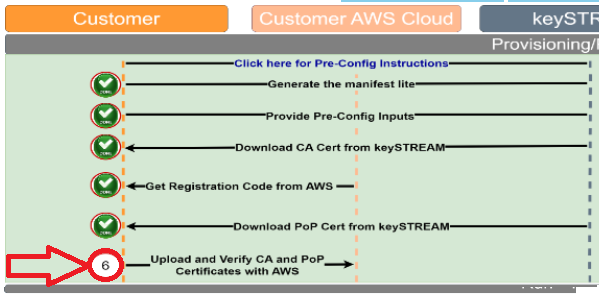
A screenshot of a computer code

Description automatically generated

After TPDS is done executing the step, the output window should have an output similar to the image above. TPDS downloads and prints the POP certificate downloaded from keySTREAM. The certificate printed is encoded with base64 encoding following ASN.1 speculations, this can be decoded with an ASN.1 decoder.

Step 2.3.4: Registering the Root CA certificate to AWS.

Click on Step 6



In this Step, TPDS will upload the downloaded CA certificate and the POP (Proof of possession) certificate to AWS using the credentials provided in Step 2.2. AWS will take in the certificates provided; it checks the presence of Registration code in the POP certificate. Once AWS verifies the presence of the correct registration code, it will use the public key of CA certificate to verify the authenticity of the POP certificate by validating the POP certificate signature. On successful completion of the steps above AWS will load the new CA certificate associated with the Fleet profile UID into the AWS account.

On successful completion of the step TPDS will print the following output.

A close-up of a computer screen

Description automatically generated

For people interested, you can check the uploaded CA certificate on the AWS UI.

1. Login to AWS
2. Go to IOT Core
3. Manage ->Security.
4. Manage->Security->Certificate Authorities
5. All certificates uploaded will be shown under CA certificate registrations.

## Results:

On successful completion of the steps in TPDS, we should be able to generate and upload manifest to your Kudelski KeySTREAM account, also we should be able to successfully upload the CA cert of the fleet profile to AWS.

## Summary:

With these steps above we achieved the following things

1. Got used to navigating TPDS UI
2. Connected the Kit and getting the device manifest.
3. Claimed the TrustManager device into KeySTREAM account using the manifest.
4. Primed TPDS with all the information required for the lab (fleet profile UID, wifi credentials, keySTREAM credentials and AWS credentials)
5. Downloaded CA certificate of our Fleet Profile UID
6. Got registration code from AWS.
7. Created a POP certificate and registered CA certificate to AWS.

With completion of the above steps, we have finished creating a product line and set up the cloud to accept new devices. Now we can get to the next lab where we will set up the embedded system and observe the output.

# Lab 3 – Setting up embedded project and observing output logs on Serial terminal and on AWS portal.

## Purpose:

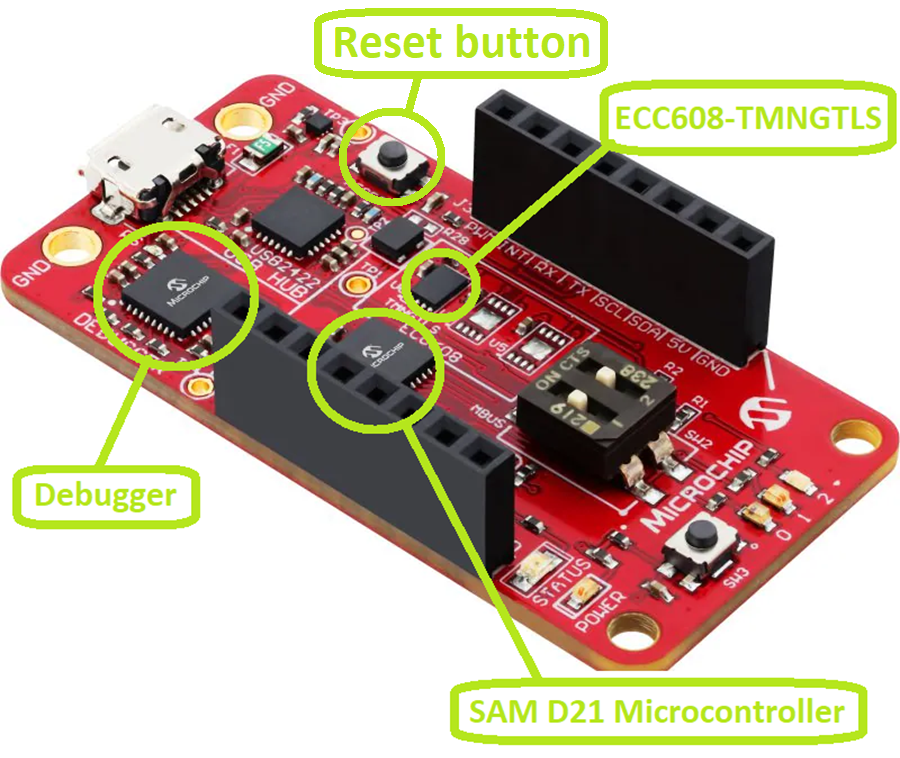
We will go through this lab and learn the embedded code structure, programming the firmware that integrates KTA and AWS to the SAM D21 microcontroller using MPLAB. Then we will observe and decode the output logs.

## Overview:

We will be doing the following steps in this lab:

1. Navigate and open the firmware project on MPLAB through TPDS
2. Learning about MPLAB UI and components
3. Accessing the embedded firmware project files
4. Understanding various sections of the firmware
5. Programming the firmware to the SAM D21 microcontroller (on the evaluation board)
6. Observing firmware output logs through serial terminal (using TeraTerm)
7. Observing messages sent to cloud under AWS web portal.

**Hardware information:**



A diagram of a computer network

Description automatically generated  
**Cryptoauth Trustmanager development board architecture for lab:**

## Procedure:

We will go through the steps to understand and get our firmware loaded onto the MCU. Then we will observe the results as serial log pushed through MCU and a IOT device activity log available under AWS.

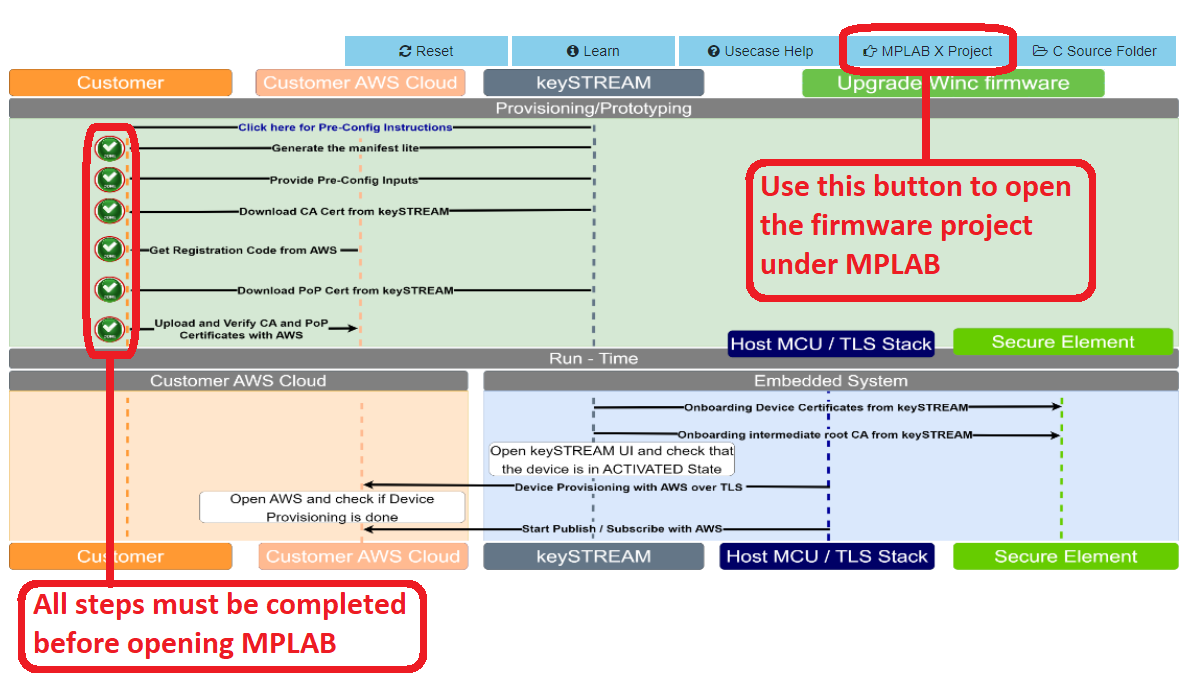
#### Step 1: Opening the firmware project with overview of MPLAB.

In this step we will see how we can open the firmware project on MPLAB and go through a brief overview of MPLAB UI itself.

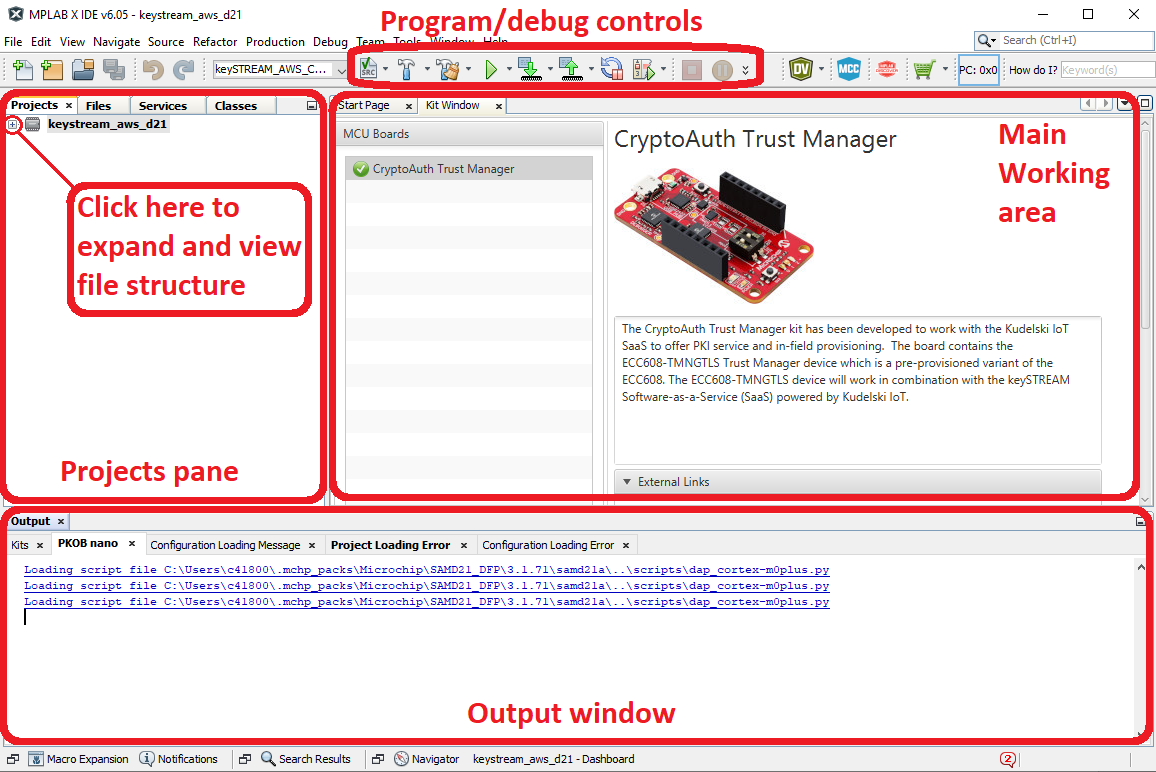
##### 1.1: Opening the firmware project on MPLAB through TPDS

TPDS software is configured in such a way that it can open MPLAB projects. In this step we will utilize TPDS to open the MPLAB project.

Whilst still on TPDS, click on the **MPLAB X Project** button. This will automatically find and open the firmware project under MPLAB. Ensure all steps within the usecase are completed before opening the MPLAB firmware project.



With that MPLAB will open and you will see a window similr to the below image. There are different panes within the IDE, some of the basics are described in the below image.

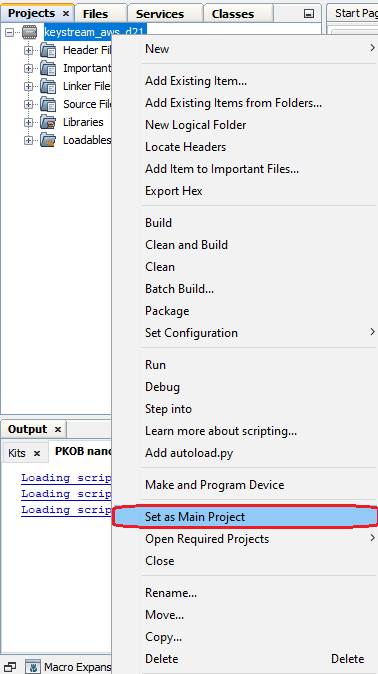


##### 1.2: MPLAB Overview

Now that we understand the basics of the IDE let us proceed to setting it up.

##### 1.2.1: Set as main project

Right click on the project and click on “Set as Main project”. The project name we are working with is “**keySTREAM\_aws\_d21**”. If the project is already set as main, you can ignore this step.



#### Step 2: Understanding and programming the embedded firmware.

##### 2.1: Accessing source files

Use the + and – buttons to navigate within the project. This is just to let users get accustomed to the UI, just for understanding the project structure.

The file cloud\_wifi\_task.c contains the function APP\_ExampleTasks(), this function has the main state machine that runs through various aspects of the project.

A screenshot of a computer

Description automatically generated

##### 2.1: Where are my project files and how project specific information is added?

Other than the project files that contain the source code for the application, we need a bit more information.

Information required by the project:

1. FleetProfileUID – Required for connection to KeySTREAM connection.
2. AWS Endpoint – To tell MCU regarding which endpoint the messages are to be posted.
3. Wifi SSID – Wifi SSID (Wifi name) to connect to Wifi network.
4. Wifi Password – Wifi SSID (Wifi name) to connect to Wifi network.

But if you have noticed, we never had to specify this information anywhere within the embedded project. This is because TPDS created a file with all these details and stored it under the following location, this file was created when clicke d on Step 2 of the TPDS usecase transaction diagram.

A blue circle with a black letter in it

Description automatically generated**Location of project files on disk**

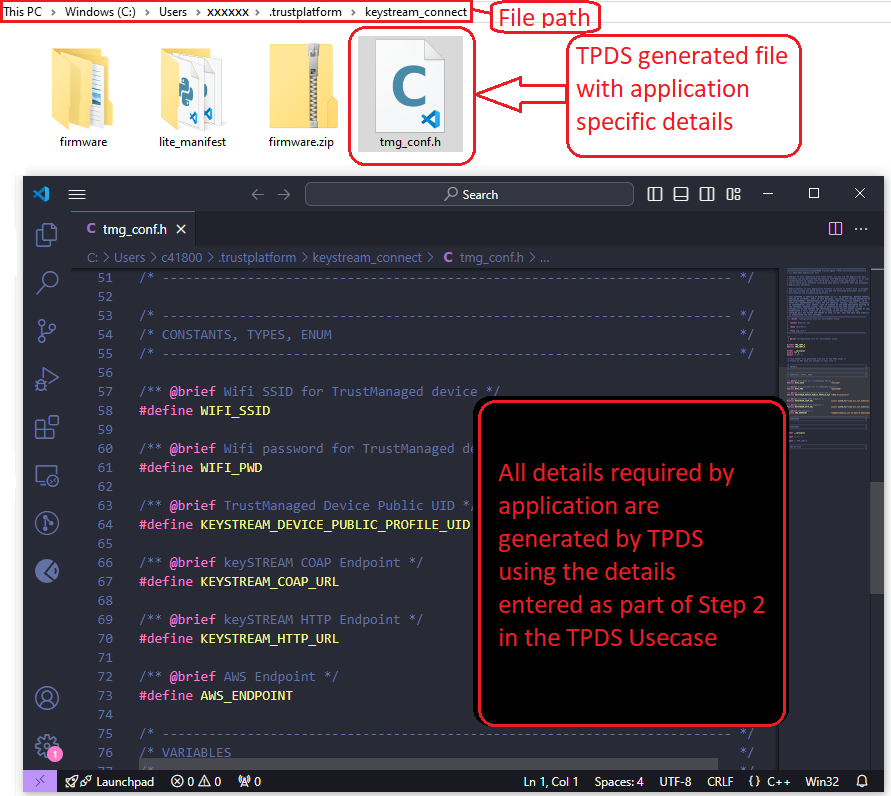
C:\Users\XXXXX\.trustplatform\keySTREAM\_connect

A blue circle with a black letter in it

Description automatically generated**Location of file TPDS created with the application level details**

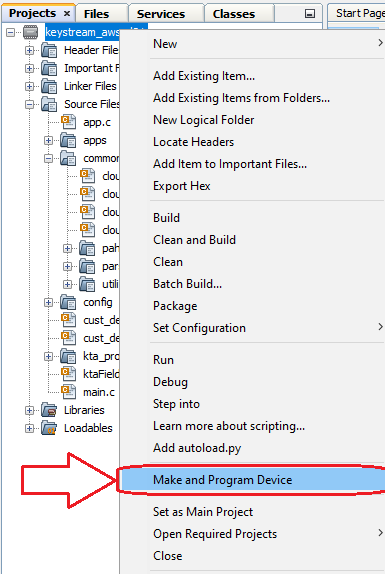
C:\Users\XXXXX\.trustplatform\keySTREAM\_connect\tmg\_conf.h

Feel free to open and explore the tmg\_conf.h. Please do not change any details, this is an auto generated file.



##### 2.2: Programming the firmware onto the Microcontroller

We can now compile and program the firmware using the “Make and program” button. This button is available when you right click on the project.

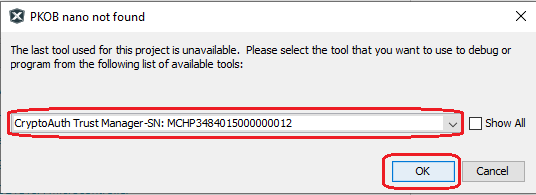


This button will compile and program the firmware project to the SAM D21 Microcontroller.

You might get the following error, this happens as the dev board information used to create the project is still retained but every dev kit has a different serial number, hence the error. This is easily solvable. Go to the dropdown menu and select the dev kit and hit “Ok”.

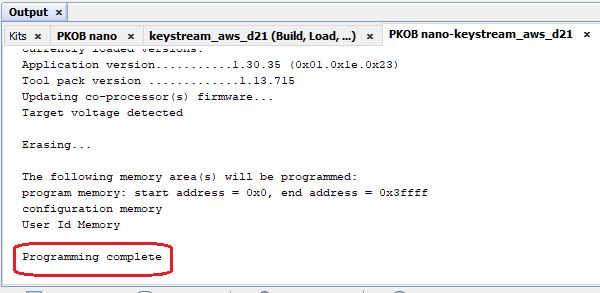
A screenshot of a computer error

Description automatically generated



Now keep a close eye on the MPLAB output window, MPLAB will give compile and programming status messages. This process taking a couple minutes is completely normal, this will depend on system performance, project size, output binary size and/or other factors.

Once the programming is complete you will see the following output message.



We now have the firmware loaded to the microcontroller. We will be looking at output logs from the microcontroller in the next steps.

##### 2.3: Observing output on serial console (TeraTerm)

The microcontroller uses the UART peripheral to print the output logs, this can then be visualized on the PC using software like TeraTerm.

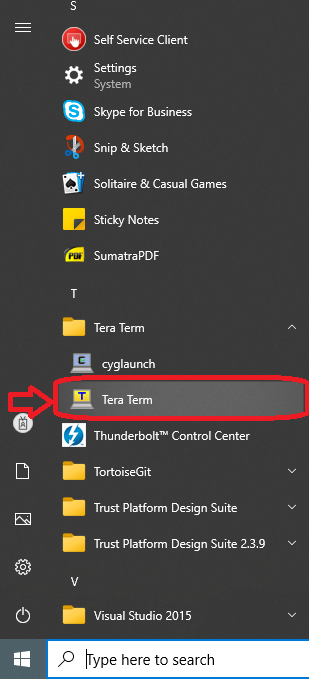
Micrcontroller uses the following serial configuration:

|  |  |
| --- | --- |
| Buadrate | 115200 |
| Data bits | 8 |
| Parity | None |
| Stop bits | 1 |
| Flow control | None |

The same settings must be configured in Tera term to read the output.

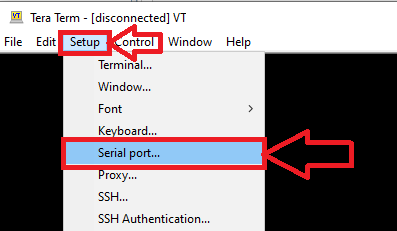
Step 2.3.1: Open TeraTerm

Under Start menu in windows navigate to Tera Term folder and click on “Tera Term” application.

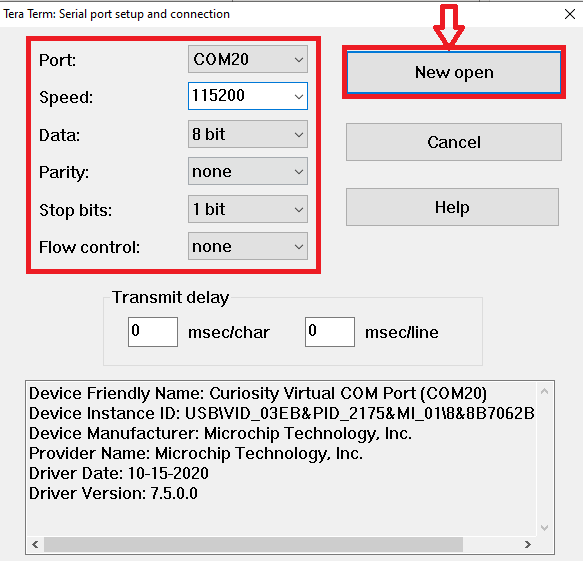


Step 2.3.2: Configure Tera Term

Go to Setup->Serial port…



Configure the Tera Term serial as shown in the below image and click on “**New open**”.



Step 2.3.3: Observing serial output on TeraTerm

Once serial port is opened you should start seeing logs coming from the microcontroller, if you do not see any output click on the reset button available on the board.

Understanding output logs can be difficult as the information is very verbose. Let us understand the output segment by segment.

Let us look at the first set of lines, here is what is happening.

* + - * Print firmware build information
      * Print Secure element information, AWS endpoint info.
      * Initialize KTA (keySTREAM Trusted Agent)
      * Connect to Wi-Fi
      * Get an IP through DHCP

A screenshot of a computer

Description automatically generated

Here is what the next set of outputs mean:

* Get in touch with keySTREAM servers.
* Get IP of AWS using DNS service (Domain name Service)
* Send information of device
* keySTREAM validates the device and starts onboarding the device.
* keySTREAM then creates a custom certificate for this specific device and starts provisioning the certificates to secure element device.
* Once this process is done, the device is considered fully onboarded.
* Again, onboarded in this case means that device is accepted by keySTREAM service and that the certificates generated for this device on keySTREAM servers are provisioned into the secure element.

A screenshot of a computer program

Description automatically generated

Then it prints the certificates that it was provisioned with

A screenshot of a computer code

Description automatically generated

Now the device has the certificates it needs to connect to the AWS service. That is exactly what comes next.

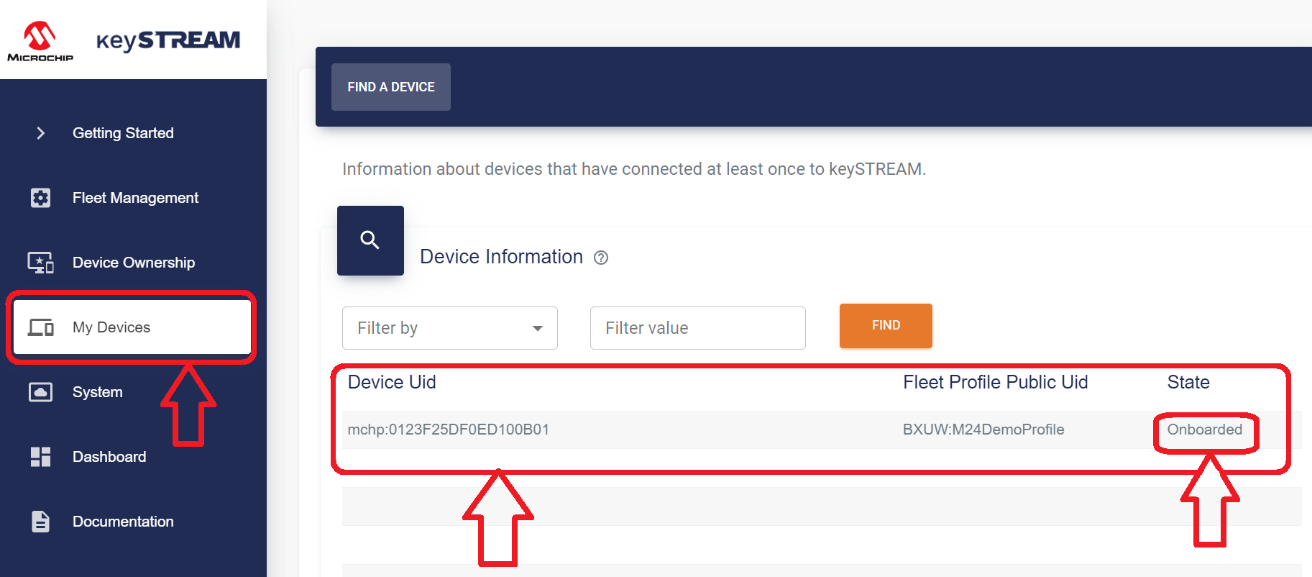
* Connect to AWS using the new certificates provisioned by keySTREAM service.
* Establish the MQTT connection.
* Subscribe to the MQTT topic
* Start pushing in MQTT messages to AWS cloud.
* The mqtt messages are pushed in every 5 seconds, the LED state is toggled, and the state of the LED is pushed to AWS.
* Then polls the keySTREAM to synchronize, checking for events.

A screenshot of a computer

Description automatically generated

Step 2.4: Observing the device on KeySTREAM UI

Go to your keySTREAM UI and go to the “My devices” tab. Here you will see your device listed, you will also see the status of the device.

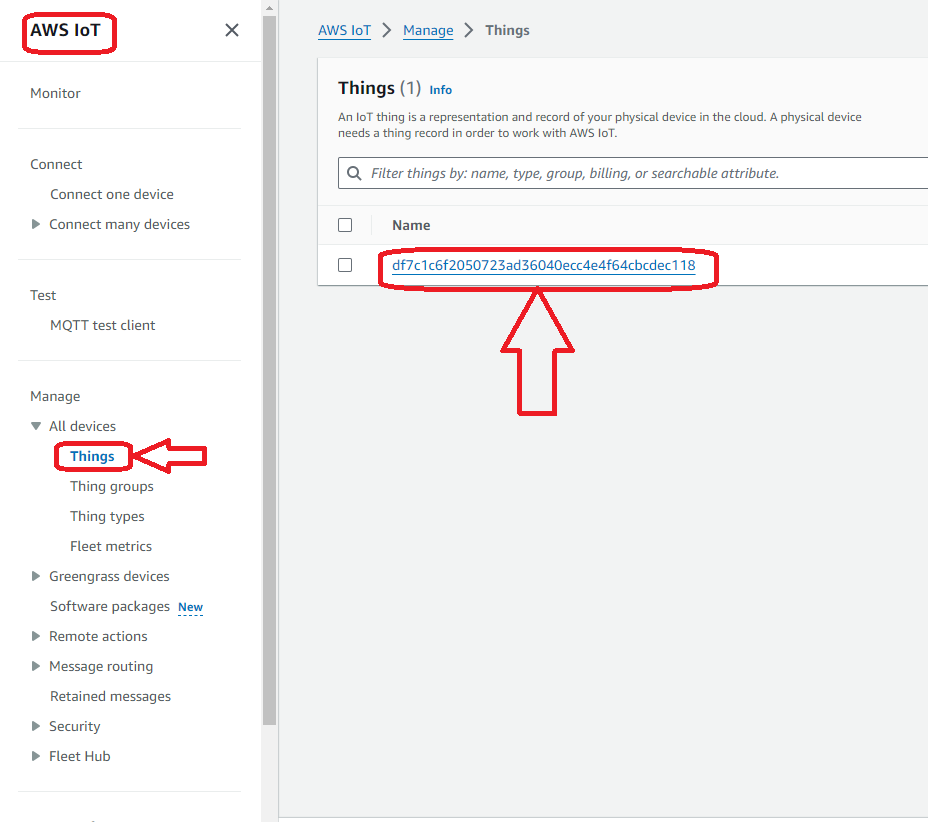


The onboarded status means that the device has contacted keySTREAM service and received new certificates.

Step 2.5: Observing the Output on AWS Web UI

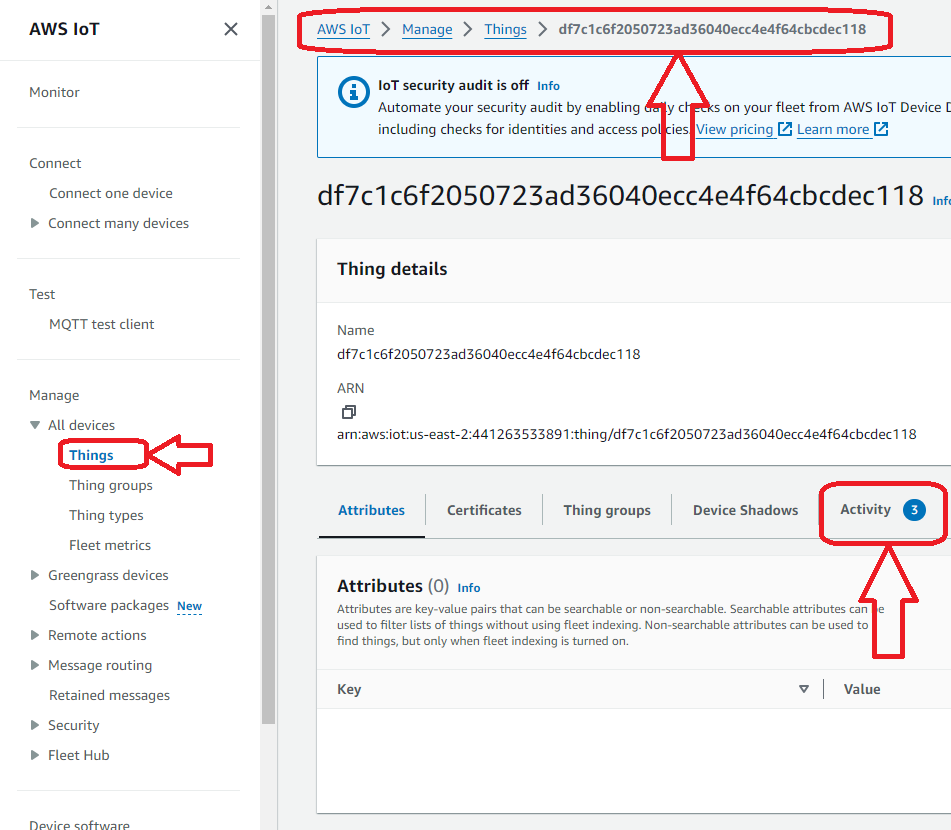
Now our device is actively posting MQTT messages to AWS, meaning you can log into your AWS account and look at these messages.

To view these messages, go to IOT Core->All devices->Things. Everyone will have a different thing name as every IOT device is unique from a certificate and identity perspective.

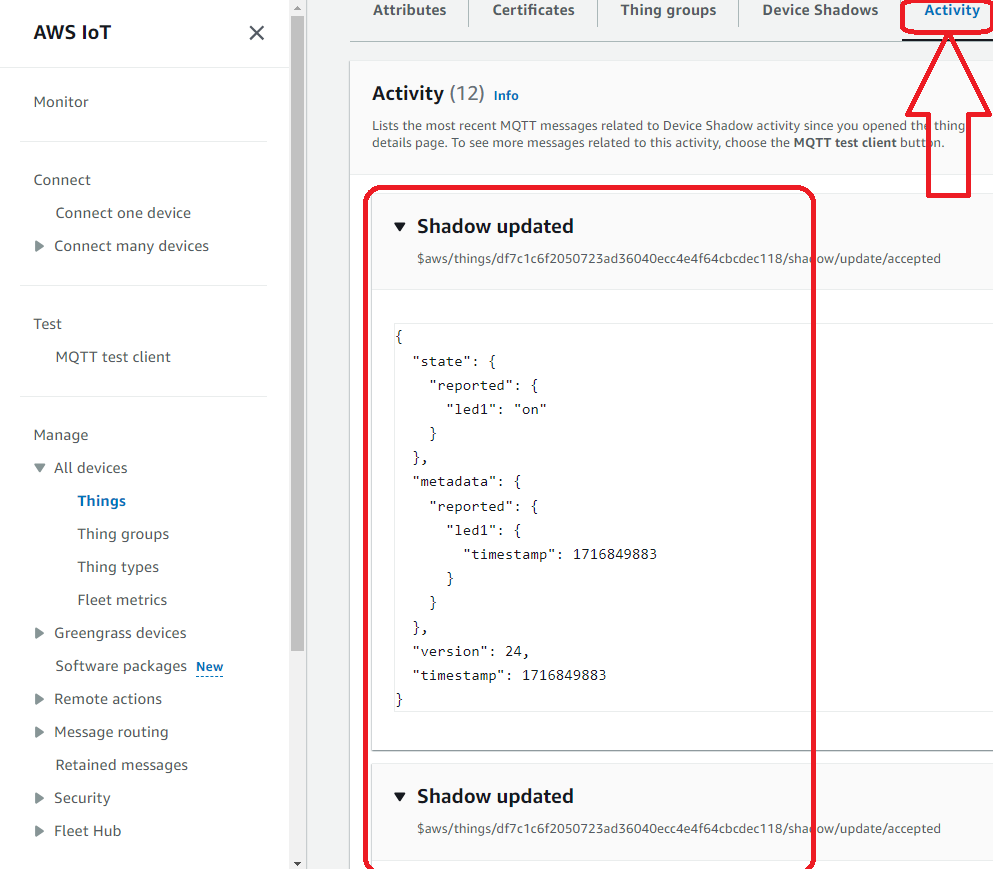


Click on the thing name, if you have multiple entries, the topmost one is usually the most recent one.

To the activities tab once you get into the thing page. There you will see messages sent by the microcontroller.



Click on activity, here you can see the messages.



## Results:

You should have been able to see messages on the serial console log, see device status on KeySTREAM portal and see MQTT messages come through and show up on AWS web portal.

## Summary:

In the lab we did the following things

1. Created keySTREAM account.
2. Created FleetProfile UID, basically setting up a product.
3. Created Auth token in KeySTREAM for API access.
4. Get device manifest using TPDS and used it to claim the device on your desk to your keySTREAM account.
5. Used TPDS to register the CA certificate associated with your fleet profile (your product) to AWS.
6. Went through various aspects of embedded firmware understanding code segments.
7. Program and observe microcontroller output logs.
8. Observed device on keySTREAM UI.
9. Observed MQTT messages through AWS web UI.

# Appendix A

## Pre-requisite configuration steps in TPDS before running the use-case.

**Note: “No action required – All these steps are completed already for MASTERs."**

1. Ensure **MPLAB X Path** is set in TPDS under **File** -> **Preference** -> **System Settings**.  
   This setup facilitates:
   * Programming the Use Case prototyping kit to factory reset application by TPDS.
   * Opening the embedded project of the use case.
2. WINC Firmware upgrade:
   * Make sure the wireless module (Wifi 7 Click - WINC1500) used in this usecase has updated firmware, this ensures the module runs the latest firmware with updated CA certs.
   * The firmware can be upgraded by clicking the **Upgrade Winc firmware** button on the usecase transaction diagram, the upgrade process can take around 12mins to 15 minutes.

# Appendix B

## AWS Setup Requirements

**Note: “No action required – All these steps are completed already for MASTERs."**

### AWS Account Setup Instructions:

In order to run the AWS Cloud Connect Use Cases, an AWS account is required. This document describes the steps required to obtain and configure an AWS account for the demo.

[Amazon Web Services (AWS)](https://aws.amazon.com/) provides computing services for a fee. Some are offered for free on a trial or small-scale basis.  
By signing up for your own AWS account, you are establishing an account to gain access to a wide range of computing services.

Think of your AWS account as your root account to AWS services. It is very powerful and gives you complete access. Be sure to protect your username and password. You control access to your AWS account by creating individual users and groups using the Identity and Access Management (IAM) Console. From the IAM Console, you also assign policies (permissions) to the group.

### Create your own AWS account:

1. Create AWS account.
   * Go to [AWS account](https://aws.amazon.com/) and follow instructions to create your own AWS account.
   * Additional details can be found at [AWS Account Creation and Activation Guide](https://aws.amazon.com/premiumsupport/knowledge-center/create-and-activate-aws-account/).
2. Secure root account with MFA (multi-factor authentication)

This is an important step to better secure your root account against attackers.  
Anyone logging in not only needs to know the password, but also a constantly changing code generated by an MFA device.

AWS recommends a number of MFA device options at the following link: [MFA device options](https://aws.amazon.com/iam/details/mfa/)

The quickest solution is a MFA app running on a phone. These apps provide the ability to scan the QR code which AWS will use to set up the MFA device.

* + Return to [AWS account](https://aws.amazon.com/) and click the **Sign In to the Console**
  + If it asks for an IAM username and password, select the **Sign-in using root account credentials** link.
  + Enter the email and password for your AWS account.
  + Under **Find Services** search for **IAM** and select it to bring up the Identity and Access Management options.
  + Click on **Activate MFA (Multi-factor Authentication) on your root account.**

1. Create an admin IAM user AWS best practices recommend not using your root account for standard administrative tasks,  
   but to create a special admin user for those tasks. See [Best Practices for Securing AWS IAM Credentials](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#lock-away-credentials)
2. Follow the instructions at [Creating an AWS IAM Admin Group](https://docs.aws.amazon.com/IAM/latest/UserGuide/getting-started_create-admin-group.html) for creating an admin user.
3. Enable MFA (multi-factor authentication) for the admin user.  
   See [AWS IAM Best Practices: Enable MFA for Privileged Users](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#enable-mfa-for-privileged-users)

**AWS CLI**

* AWS CLI is already installed during the TPDS installation. You do not need to worry about it.

**Configuring the account using CloudFormation Templates**

The usage of a custom PKI with TrustMANAGER device uses AWS IoT Core and does not require AWS JITR(Just In Time Registration).  
AWS IOT requires few resources to be setup with an AWS account to work. The creation of these resources is automated leveraging the AWS CloudFormation service. The setup takes about 3-4 minutes.

1. Sign into the [AWS console](https://aws.amazon.com/) using the admin user created in the previous section.
2. Change to region to **US East (Ohio)** (a.k.a. us-east-2). This is done from a dropdown in the top right of the console webpage after logging in.
3. Under **Find Services** search for **CloudFormation** and select it to bring up that service.
4. Click **Create Stack** button.
5. Select **Upload a template file** from the page of the stack creation.
6. Click **Choose file** and upload the **aws-zero-touch-full-setup.yaml** file. Note, if running from a China region, you will need to select the  
   **aws-zero-touch-full-setup-cn.yaml** instead. These files are available in ~/.trustplatform folder.
7. Click **Next** to move on to the stack details.
8. Enter **TrustMANAGER** as the stack name. Actual name isn't important, just has to be unique.
9. Enter a password for the user that will be created to run the demo under **UserPassword**. It's important the password has small characters,  
   capitals, numbers and at least 1 special character.
10. Click **Next** to move on to the stack options. Nothing needs to be changed here.
11. Click **Next** to move on to the stack review.
12. Check the acknowledgement box regarding IAM resources at the bottom of the page.
13. Click **Create Stack** to start the resource creation.
14. Wait until the stack creation is complete. This can take a few minutes. Once done, the stack you created will show as CREATE\_COMPLETE.
15. Save demo credentials. Click the **Outputs** tab for the stack to see the credentials to be saved. The Outputs tab is where you will find the **AccessKey** and **SecretAccessKey** TPDS will ask you to enter in the next step.