ECC608 AES Message Encryption

TPDS Usecase Guide

Table of Contents

ECC608 AES Message Encryption

This application example demonstrates AES encryption being run on Host MCU or MPU while having the master symmetric key held securely in ECC608 secure element.

Description

- The master symmetric key is stored in ECC608 and a derived key is generated using KDF command. The parameters used to calculate the derived key are then shared to the Cloud/ remote host so it can calculate the same derived key to perform AES operations.
- Storing the symmetric key in ECC608 ensures that the master key is never exposed.
- The derived key can also be set to expire (ephemeral key) after a set time frame in the software.
- Once the current key expires, the remote host and MCU/MPU system can agree on parameters and generate a fresh ephemeral key.

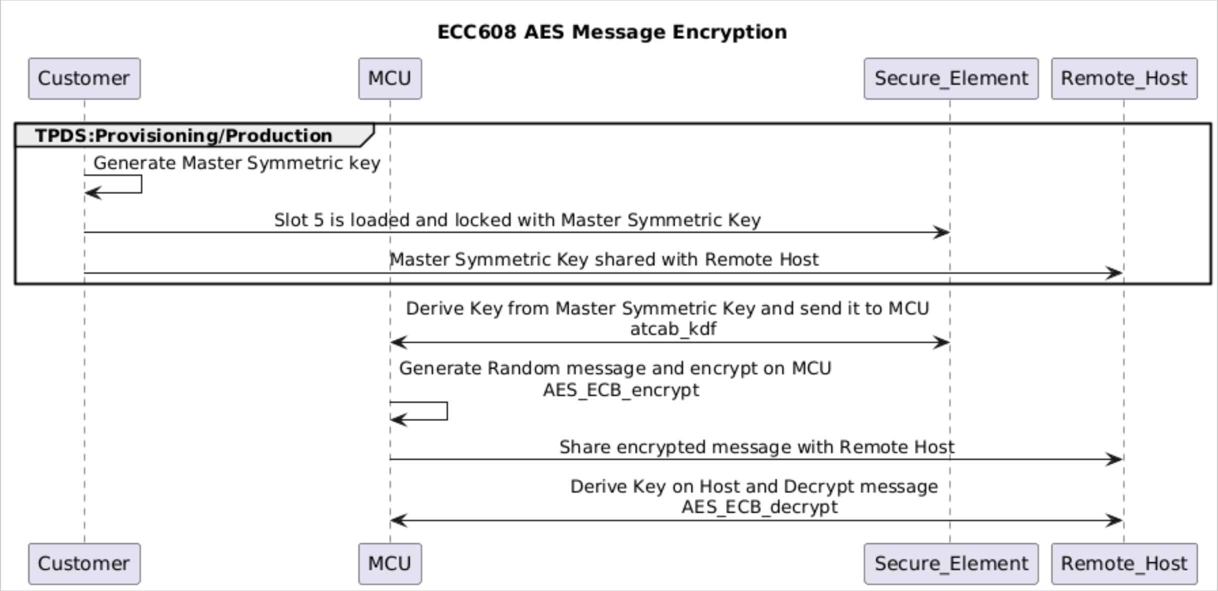

ECC608 AES Message Encryption
Customer
MCU
Secure_Element
Remote_Host
TPDS:Provisioning/Production
Generate Master Symmetric key
Slot 5 is loaded and locked with Master Symmetric Key
Master Symmetric Key shared with Remote Host
Derive Key from Master Symmetric Key and send it to MCU
atcab_kdf
Generate Random message and encrypt on MCU
AES_ECB_encrypt
Share encrypted message with Remote Host
Derive Key on Host and Decrypt message
AES_ECB_decrypt
Customer
MCU
Secure_Element
Remote_Host

Figure-1

Training Video

Microchip Security
Part I - Cryptography Primer

Figure-2

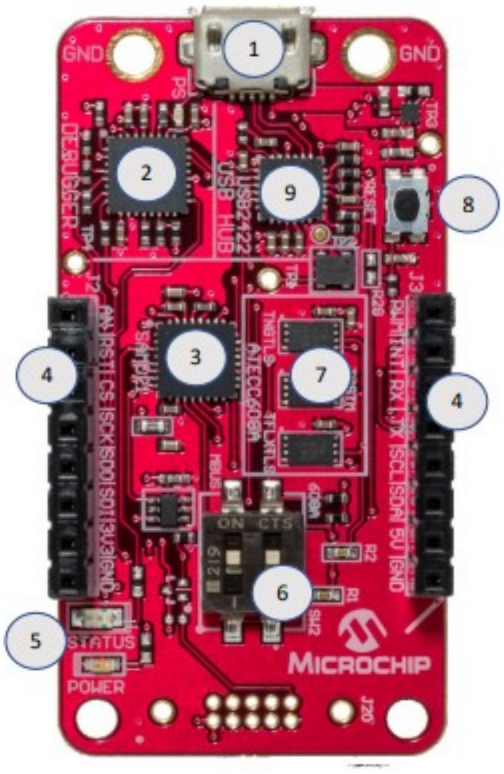
ATECC608 AES Message Encryption Execution

Prerequisites

- Trust Platform Design Suite
- MPLAB® X IDE
- Cryptoauth Trust Platform Development Kit

Setting up Cryptoauth Trust Platform Development Kit

- Ensure both the ON and CTS switches are in the ON position in the Dual SPST DIP Switch.



1. USB Connector
2. nEDBG On-board Debugger
3. ATSAMD21E18A Host MCU
4. mikroBUS™ Headers
5. Power and Status LEDs
6. Dual SPST DIP Switch
7. ATECC608A Secure Elements
8. Reset Switch
9. USB2422 Dual Port Hub

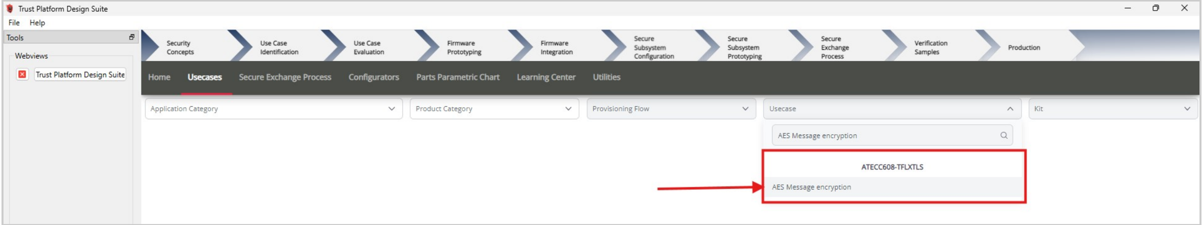
Figure-3

- Connect the micro USB port on the board to the computer using a micro USB cable.

Opening the ECC608 AES Message Encryption Use Case

- Open Trust Platform Design Suite and navigate to Usecases Section.

- In the Use Case dropdown,search for "AES Message Encryption" and select "AES Message encryption" under the ATECC608-TFLXTLS group.



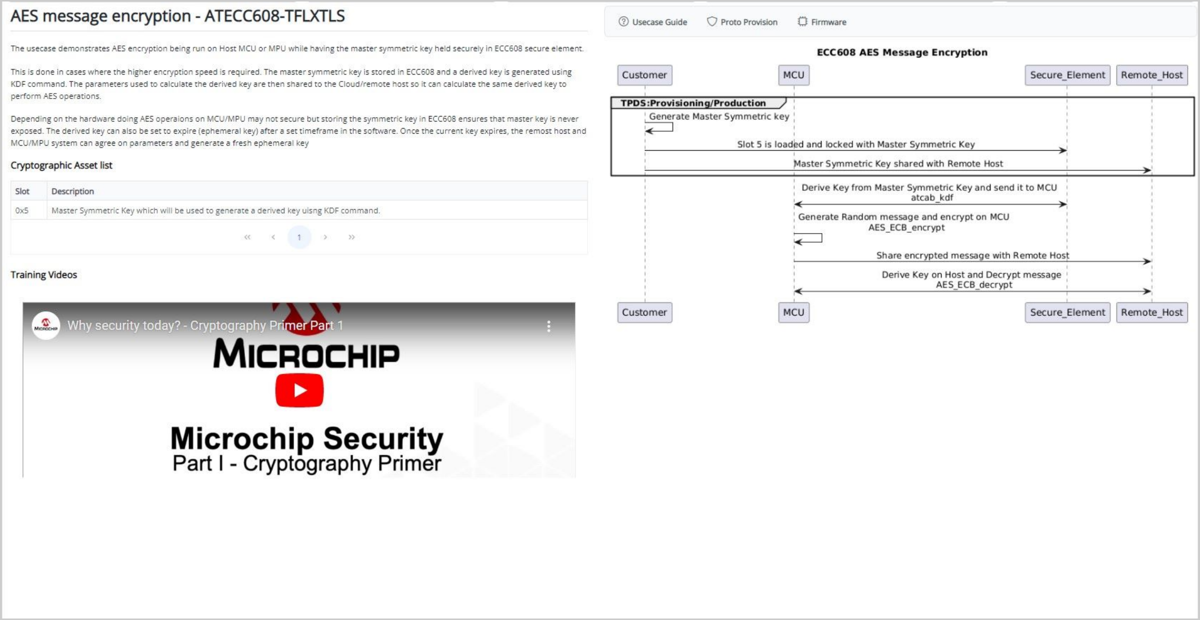
AES Message encryption

ATECC608-TFLXTLS

AES Message encryption

Figure-4

- The ECC608 AES Message Encryption use case will open.



AES message encryption - ATECC608-TFLXTLS

The usecase demonstrates AES encryption being run on Host MCU or MPU while having the master symmetric key held securely in ECC608 secure element.

This is done in cases where the higher encryption speed is required. The master symmetric key is stored in ECC608 and a derived key is generated using KDF command. The parameters used to calculate the derived key are then shared to the Cloud/remote host so it can calculate the same derived key to perform AES operations.

Depending on the hardware doing AES operaions on MCU/MPU may not secure but storing the symmetric key in ECC608 ensures that master key is never exposed. The derived key can also be set to expire (ephemeral key) after a set timeframe in the software. Once the current key expires, the remost host and MCU/MPU system can agree on parameters and generate a fresh ephemeral key

Cryptographic Asset list

Slot	Description
0x5	Master Symmetric Key which will be used to generate a derived key uisng KDF command.

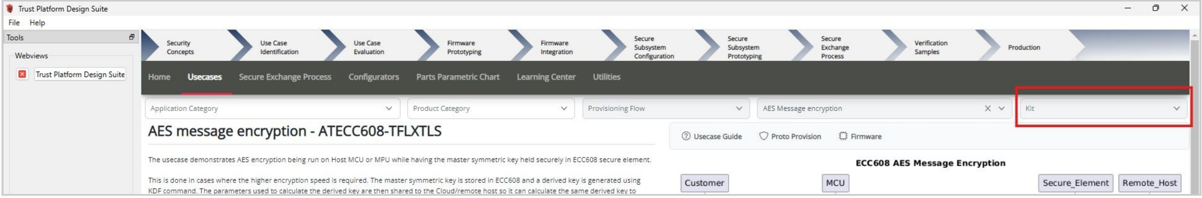
Training Videos

Why security today? - Cryptography Primer Part 1
Microchip
Microchip Security
Part I - Cryptography Primer
Usecase Guide
Proto Provision
Firmware
ECC608 AES Message Encryption
Customer
MCU
Secure_Element
Remote_Host
TPDS:Provisioning/Production
Generate Master Symmetric key
Slot 5 is loaded and locked with Master Symmetric Key
Master Symmetric Key shared with Remote Host
Derive Key from Master Symmetric Key and send it to MCU
atcab_kdf
Generate Random message and encrypt on MCU
AES_ECB_encrypt
Share encrypted message with Remote Host
Derive Key on Host and Decrypt message
AES_ECB_decrypt

Figure-5

Provisionig Usecases Resources

- From the Kit Dropdown, select the Cryptoauth Trust Platform Kit.



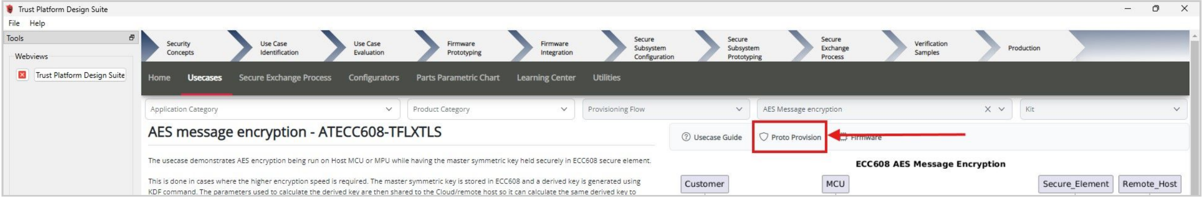
AES message encryption - ATECC608-TFLXTLS

The usecase demonstrates AES encryption being run on Host MCU or MPU while having the master symmetric key held securely in ECC608 secure element.

This is done in cases where the higher encryption speed is required. The master symmetric key is stored in ECC608 and a derived key is generated using KDF command. The parameters used to calculate the derived key are then shared to the Cloud/remote host so it can calculate the same derived key to

Figure-6

- Click on Proto Provision.



AES message encryption - ATECC608-TFLXTLS

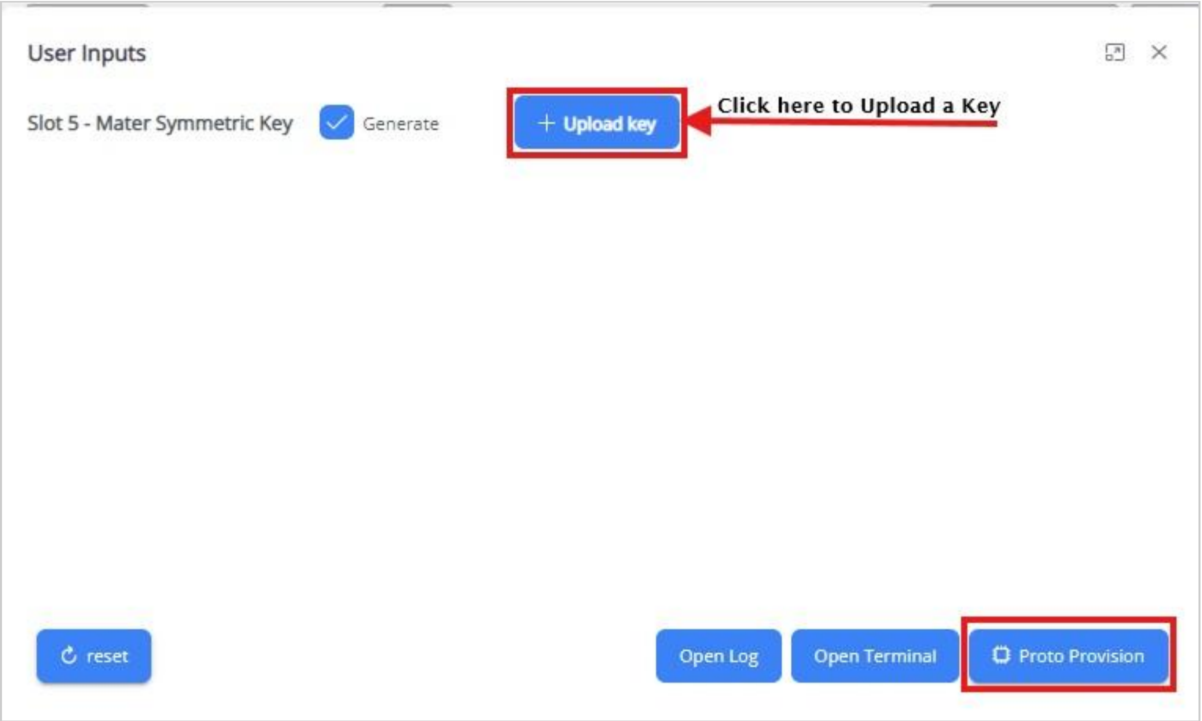
The usecase demonstrates AES encryption being run on Host MCU or MPU while having the master symmetric key held securely in ECC608 secure element.

This is done in cases where the higher encryption speed is required. The master symmetric key is stored in ECC608 and a derived key is generated using KDF command. The parameters used to calculate the derived key are then shared to the Cloud/remote host so it can calculate the same derived key to

Figure-7

- Select the Generate option to create and use a new Symmetric Key, or upload a user-specific Symmetric Key for Mater Symmetric Key.

- Click on Proto Provision



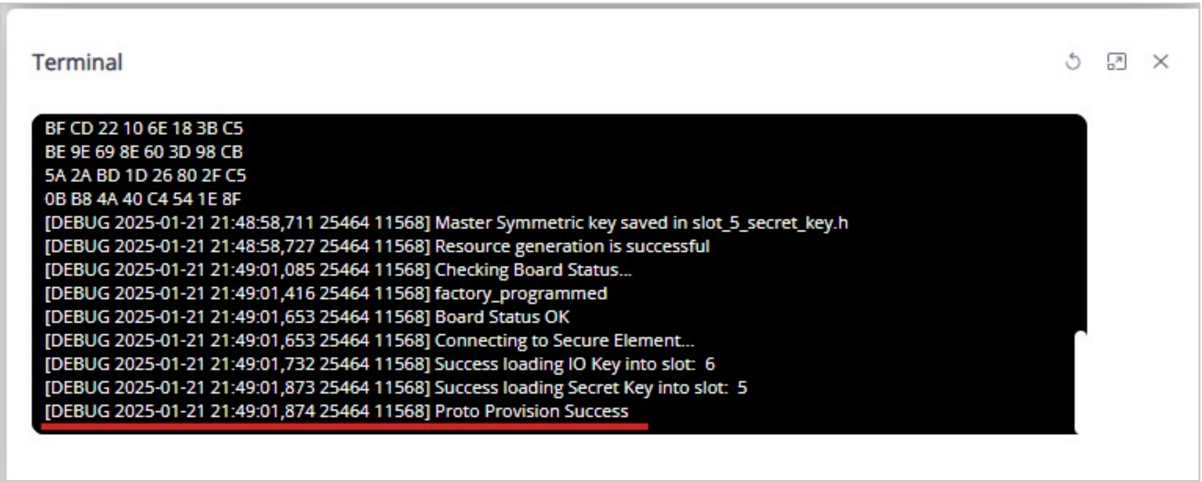
Slot 5 - Mater Symmetric Key ☑ Generate

Click here to Upload a Key

Figure-8

- The necessary resources will be created at `~/.trustplatform/aes_message_encryption`:
  - **slot_5_secret_key.h**: This file contains a 32-byte Master symmetric key, which is loaded into Slot 5, and it will be utilized to perform kdf operation to derive key in the Firmware project.
  - **slot_5_secret_key.pem**: This file contains the generated symmetric key in PEM format.
  - **slot_6_secret_key.h**: This file contains the IO Protection key which is loaded into Slot 6.
  - **slot_6_secret_key.pem**: This file contains the IO Protection key which is loaded into Slot 6 in PEM format.
- Click on Yes in the pop-up to load resources onto ECC608.

- **Proto Provision Success Toast** will pop up after successfully loading resources, Proto Provision Success will be logged on to Terminal.

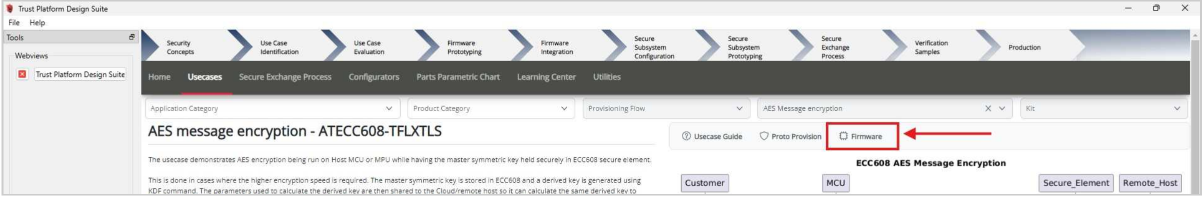


```
BF CD 22 10 6E 18 3B C5
BE 9E 69 8E 60 3D 98 CB
5A 2A BD 1D 26 80 2F C5
0B B8 4A 40 C4 54 1E 8F
[DEBUG 2025-01-21 21:48:58,711 25464 11568] Master Symmetric key saved in slot_5_secret_key.h
[DEBUG 2025-01-21 21:48:58,727 25464 11568] Resource generation is successful
[DEBUG 2025-01-21 21:49:01,085 25464 11568] Checking Board Status...
[DEBUG 2025-01-21 21:49:01,416 25464 11568] factory_programmed
[DEBUG 2025-01-21 21:49:01,653 25464 11568] Board Status OK
[DEBUG 2025-01-21 21:49:01,653 25464 11568] Connecting to Secure Element...
[DEBUG 2025-01-21 21:49:01,732 25464 11568] Success loading IO Key into slot:  6
[DEBUG 2025-01-21 21:49:01,873 25464 11568] Success loading Secret Key into slot:  5
[DEBUG 2025-01-21 21:49:01,874 25464 11568] Proto Provision Success
```

Figure-9

Build and Program Application

- Once the resources have been successfully loaded, open the Firmware Project by clicking on the Firmware button.



AES message encryption - ATECC608-TFLXTLS

The usecase demonstrates AES encryption being run on Host MCU or MPU while having the master symmetric key held securely in ECC608 secure element.

This is done in cases where the higher encryption speed is required. The master symmetric key is stored in ECC608 and a derived key is generated using KDF command. The parameters used to calculate the derived key are then shared to the Cloud/remote host so it can calculate the same derived key to

Figure-10

- The project **aes_msg_enc_ecc608** will open in the MPLABX IDE.
- Right-click on **aes_msg_enc_ecc608** and select "Set as Main Project".

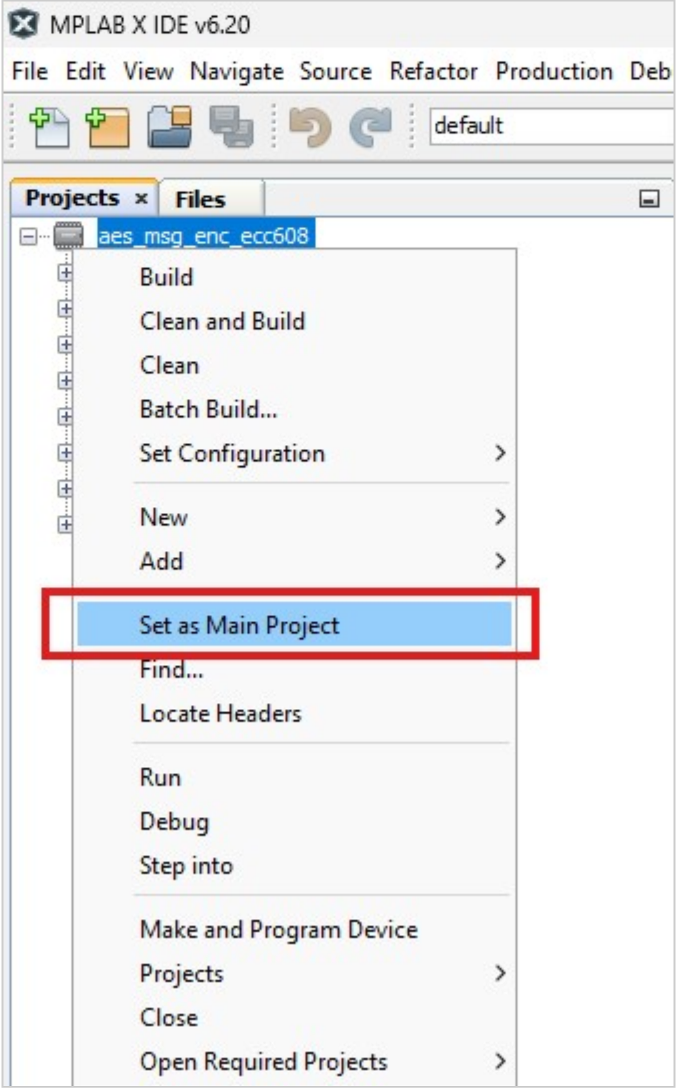


Figure-11

- Click on "Make and Program Device".

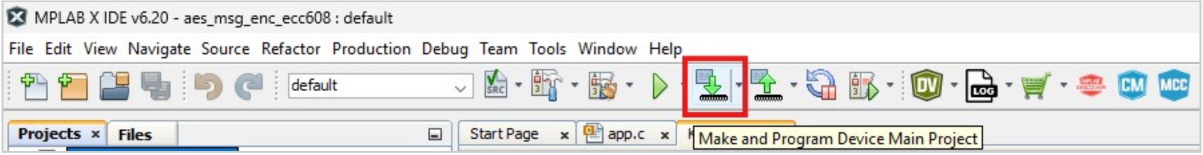
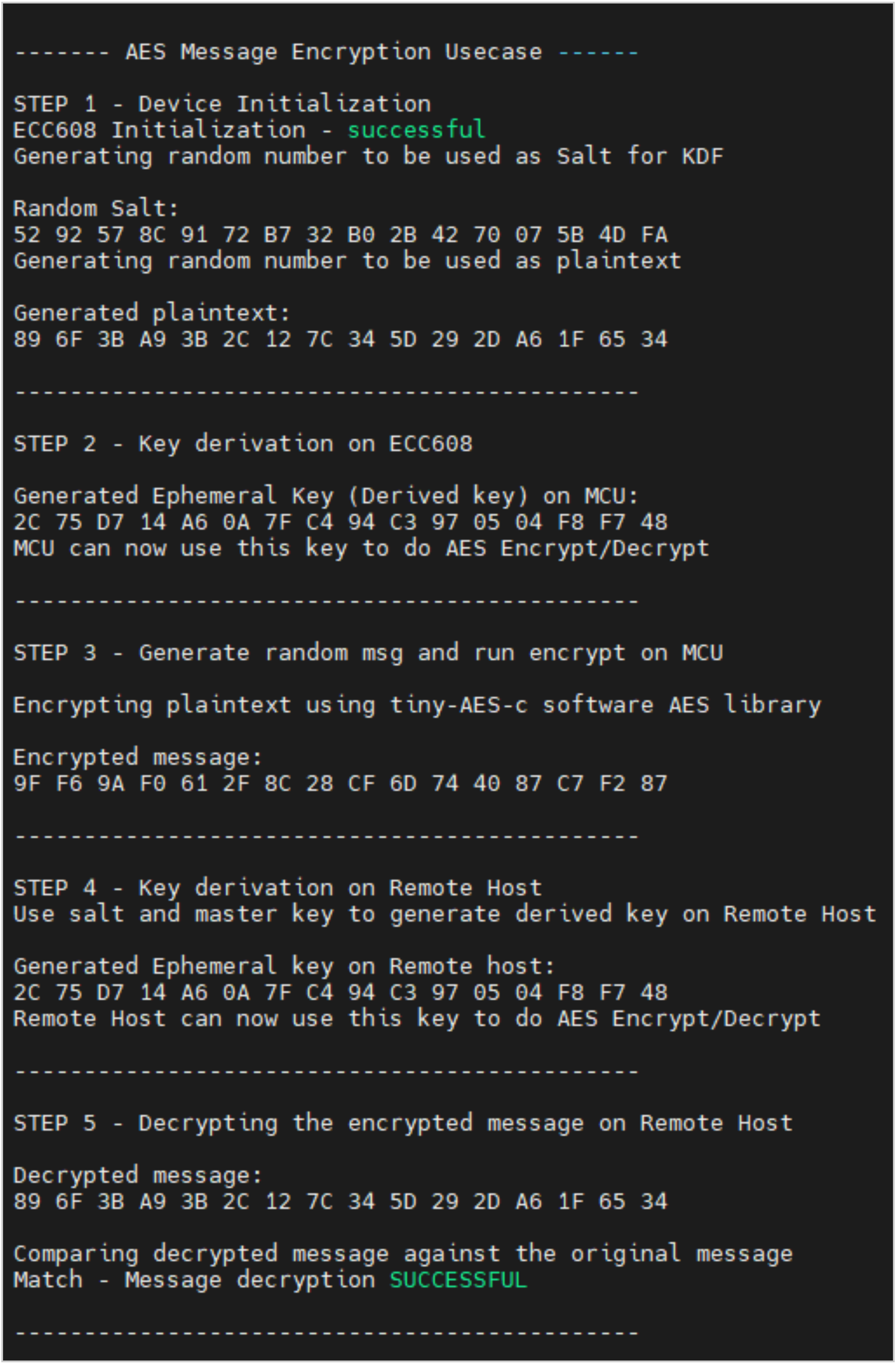

MPLAB X IDE v6.20 - aes_msg_enc_ecc608 : default
File Edit View Navigate Source Refactor Production Debug Team Tools Window Help
default
Projects
Files
Start Page
app.c
Make and Program Device Main Project

Figure-12

- After the programming process is complete, launch the Terminal application (e.g., Tera Term) on your computer.

- Connect to the Virtual COM port and configure the serial settings as follows:
  - Baud : 115200
  - Data : 8 Bits
  - Parity : None
  - Stop : 1 Bit
  - Flow Control : None
- Press the Reset button on Cryptoauth Trust Platform Development Kit
- Review the output message in the console:



```
------- AES Message Encryption Usecase ------

STEP 1 - Device Initialization
ECC608 Initialization - successful
Generating random number to be used as Salt for KDF

Random Salt:
52 92 57 8C 91 72 B7 32 B0 2B 42 70 07 5B 4D FA
Generating random number to be used as plaintext

Generated plaintext:
89 6F 3B A9 3B 2C 12 7C 34 5D 29 2D A6 1F 65 34

---------------------------------------------

STEP 2 - Key derivation on ECC608

Generated Ephemeral Key (Derived key) on MCU:
2C 75 D7 14 A6 0A 7F C4 94 C3 97 05 04 F8 F7 48
MCU can now use this key to do AES Encrypt/Decrypt

---------------------------------------------

STEP 3 - Generate random msg and run encrypt on MCU

Encrypting plaintext using tiny-AES-c software AES library

Encrypted message:
9F F6 9A F0 61 2F 8C 28 CF 6D 74 40 87 C7 F2 87

---------------------------------------------

STEP 4 - Key derivation on Remote Host
Use salt and master key to generate derived key on Remote Host

Generated Ephemeral key on Remote host:
2C 75 D7 14 A6 0A 7F C4 94 C3 97 05 04 F8 F7 48
Remote Host can now use this key to do AES Encrypt/Decrypt

---------------------------------------------

STEP 5 - Decrypting the encrypted message on Remote Host

Decrypted message:
89 6F 3B A9 3B 2C 12 7C 34 5D 29 2D A6 1F 65 34

Comparing decrypted message against the original message
Match - Message decryption SUCCESSFUL

---------------------------------------------
```

Figure-13

• Console displays a message stating the decrypted message matches the original message.

Conclusion

The ECC608 AES Message Encryption use case demonstrates a secure method for encrypting messages by storing the master symmetric key within the ECC608 secure element. This ensures the key is never exposed, enhancing security. The guide provides detailed steps for setting up the hardware and software, provisioning resources, and programming the application. Successful execution and verification confirm the system's effectiveness in securely handling AES encryption. This use case highlights the importance of secure key storage and management in cryptographic operations.

Microchip Information

The Microchip Website

Microchip provides online support via our website at www.microchip.com/. This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** - Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** - Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** - Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Product Change Notification Service

Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to www.microchip.com/pcn and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the website at: www.microchip.com/support

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.

- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip product is strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable”. Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.

Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at www.microchip.com/en-us/support/design-help/client-support-services.

THIS INFORMATION IS PROVIDED BY MICROCHIP “AS IS”. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP’S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Trademarks

The Microchip name and logo, the Microchip logo, Adaptec, AVR, AVR logo, AVR Freaks, BesTime, BitCloud, CryptoMemory, CryptoRF, dsPIC, flexPWR, HELDO, IGLOO, JukeBlox, KeeLoq, Kleer, LANCheck, LinkMD, maXStylus, maXTouch, MediaLB, megaAVR, Microsemi, Microsemi logo, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, PolarFire, Prochip Designer, QTouch, SAM-BA, SenGenuity, SpyNIC, SST, SST Logo, SuperFlash, Symmetricom, SyncServer, Tachyon, TimeSource, tinyAVR, UNI/O, Vectron, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

AgileSwitch, ClockWorks, The Embedded Control Solutions Company, EtherSynch, Flashtec, Hyper Speed Control, HyperLight Load, Libero, motorBench, mTouch, Powermite 3, Precision Edge,

ISBN: 978-1-6683-0382-5

Quality Management System

For information regarding Microchip’s Quality Management Systems, please visit www.microchip.com/quality.