Renesas RA Family

RA AWS Cloud Connectivity and Firmware Update OTA on CK-RA6M5 v2 with Ethernet – Getting Started Guide

Introduction

This document presents the AWS cloud connectivity Application Project with Over the Air (OTA) firmware update on the CK-RA6M5 v2 kit, utilizing the Ethernet interface. The project encompasses three main components:

- A secure bootloader project.
- Cloud connectivity application with downloader functionality.
- AWS OTA cloud-side configurations, image storage, and OTA initiation.

This document also serves as a comprehensive guide to understanding and implementing Over-the-Air updates using AWS IoT OTA. It explores the objectives, architectural components, and scope of OTA updates, as well as provides step-by-step instructions for preparing, executing, and validating updates on the CK-RA6M5 v2 board.

Applies to:

- RA6M5 MCU Group

Required Resources

To build and run the application project, the following resources are needed.

Development tools and software

- Flexible Software Package (FSP) v6.1.0 and required tools (renesas.com/us/en/software-tool/flexible-software-package-fsp)
- Openssl: v3.0.12 or later (OpeenSSLwebsite: https://www.openssl.org/, OpenSSL App for Windows OS: https://slproweb.com/products/Win32OpenSSL.html)
- Python: v3.12.0 or later (https://www.python.org/)

Hardware

- Renesas CK-RA6M5 v2 kit (renesas.com/ra/ck-ra6m5)
- PC running Windows® 10 or Windows® 11 and an installed web browser (Google Chrome, Microsoft Edge, Mozilla Firefox, or Safari)
- Micro-B USB cables included as part of the kit. See *CK-RA6M5 v2 – User's Manual*)
- USB-C cable (See *CK-RA6M5 v2 – User's Manual)*
- CAT-5 /CAT-6 Ethernet cable
- A router/switch with at least one available 100 Mbps/full duplex Ethernet port with connectivity to the internet

Prerequisites and Intended Audience

This application note assumes that the user is adept at operating the Renesas e^2 studio IDE with Flexible Software Package (FSP). If not, we recommend reading and following the procedures in the *FSP User's Manual* sections for 'Starting Development', including 'Debug the Blinky Project'. Doing so enables familiarization with e^2 studio and FSP and validates proper debug connection to the target board. In addition, this application note assumes prior knowledge of Cloud connectivity and its communication protocols, and knowledge of firmware upgrades over the air.

The intended audience is users who want to develop Firmware Update OTA applications with MQTT/TLS modules using Ethernet modules on Renesas RA6 MCU Series.

Note: If you are a first-time user of e² studio and FSP, we highly recommend you install e² studio and FSP on your system to run the Blinky Project and to get familiar with the e² studio and FSP development environment before proceeding to the next sections.

Note: This Application Project and Application Note are guaranteed to work only with FSP v6.1.0

Prerequisites

1. Access to online documentation is available in the **References** section.
2. Knowledge of the MCU Bootloader and access to the Renesas Bootloader documentation provided with FSP.
3. Knowledge of OTA and access to the AWS OTA documentation.
4. Access to the latest documentation for the identified Renesas Flexible Software Package.
5. Prior knowledge of operating the e² studio and the built-in (or standalone) RA Configurator.
6. Access to associated hardware documentation, such as User Manuals, Schematics, and other relevant kit information (renesas.com/ra/ck-ra6m5).

Contents

1. Firmware Update Over the Air

1.1 Introduction

Firmware update is a critical aspect of modern device management, allowing seamless and efficient deployment of software updates to connected devices. Leveraging the capabilities of AWS OTA, organizations can streamline the process of updating firmware and software on their IoT devices remotely.

By harnessing AWS OTA, organizations can ensure their IoT devices remain up to date with the latest features, security patches, and improvements, all while minimizing downtime and operational disruptions. In the following sections, we will explore key features of the RA OTA solution, its benefits, and its Architecture.

1.2 Overview of Renesas RA AWS IoT OTA

RA OTA solution is a comprehensive platform meticulously designed to streamline the process of updating firmware for Renesas microcontroller-based devices, prioritizing both security and efficiency. Through seamless integration with AWS IoT, the RA OTA solution harmoniously blends with FSP (Flexible Software Package), empowering users to effortlessly configure and establish connections to the AWS Cloud, thereby facilitating smooth firmware updates. Additionally, the RA OTA solution boasts a robust, secure bootloader, which serves as a safeguard, ensuring the integrity and safety of firmware updates. In essence, the RA OTA solution represents a holistic solution, addressing the complexities of both cloud connectivity and firmware updates and offering a seamless and dependable experience for users.

1.2.1 Objectives

- Enable efficient and reliable OTA updates for IoT devices equipped with a bootloader and downloader, streamlining the update process and minimizing downtime.
- Ensure secure delivery of firmware updates to IoT devices by leveraging encryption, authentication, and access control mechanisms in the bootloader and downloader components.
- Implement a robust bootloader component capable of managing the update process, including firmware verification, update package validation, and rollback procedures in case of update failures.
- Manage multiple versions of firmware updates, enabling seamless transition between different firmware versions and ensuring compatibility with existing hardware configurations.
- Integrate with other AWS services, such as AWS IoT Core, AWS Lambda, and Amazon S3, to leverage features such as device provisioning, message brokering, and cloud storage, enabling end-to-end IoT solution deployment and management.

1.2.2 Key Features

RA OTA solution encompasses several key features aimed at delivering a robust and reliable firmware update solution:

- RA OTA solution seamlessly integrates with AWS IoT, providing users with a secure and scalable platform for managing firmware updates. This integration enables seamless communication between Renesas devices and the AWS Cloud, facilitating efficient Over-The-Air updates.
- Flexible Software Package (FSP) Compatibility: RA OTA solution is fully compatible with Renesas' Flexible Software Package (FSP), ensuring seamless integration with existing development workflows. This compatibility allows developers to leverage the RA OTA solution without significant modifications to their existing projects.
- Secure Bootloader: Security is paramount in firmware updates, and the RA OTA solution addresses this by incorporating a secure bootloader into its platform. The secure bootloader ensures the authenticity and integrity of firmware updates, mitigating the risk of unauthorized access or tampering.
- Over-The-Air Configuration: RA OTA solution provides users with the flexibility to configure and manage firmware updates Over-The-Air. This feature allows for remote scheduling, deployment, and monitoring of firmware updates, empowering users to maintain control over their devices' software lifecycle.

1.2.3 Benefits

RA OTA solution offers a range of benefits to developers, device manufacturers, and end-users alike:

- Enhanced Efficiency: By automating the firmware update process and enabling Over-The-Air updates, the RA OTA solution reduces the time and effort required to deploy updates across a fleet of devices, thereby increasing operational efficiency.

- Improved Security: With its secure bootloader and integration with AWS IoT, the RA OTA solution enhances the security of firmware updates, safeguarding devices against unauthorized access and potential security threats.
- Seamless Integration: RA OTA solution seamlessly integrates with existing development workflows and Renesas' FSP, minimizing disruptions to the development process and enabling a smooth transition to Over-The-Air firmware updates.
- Enhanced User Experience: By providing a user-friendly interface and intuitive Over-The-Air update capabilities, the RA OTA solution enhances the overall user experience, making firmware management simple and accessible for developers and end-users.

1.2.4 Architecture

The architecture of the RA OTA solution is designed to provide a robust and scalable framework for managing firmware updates for Renesas microcontroller-based devices. This section will delve into the key components and the flow of data within the RA OTA solution architecture.

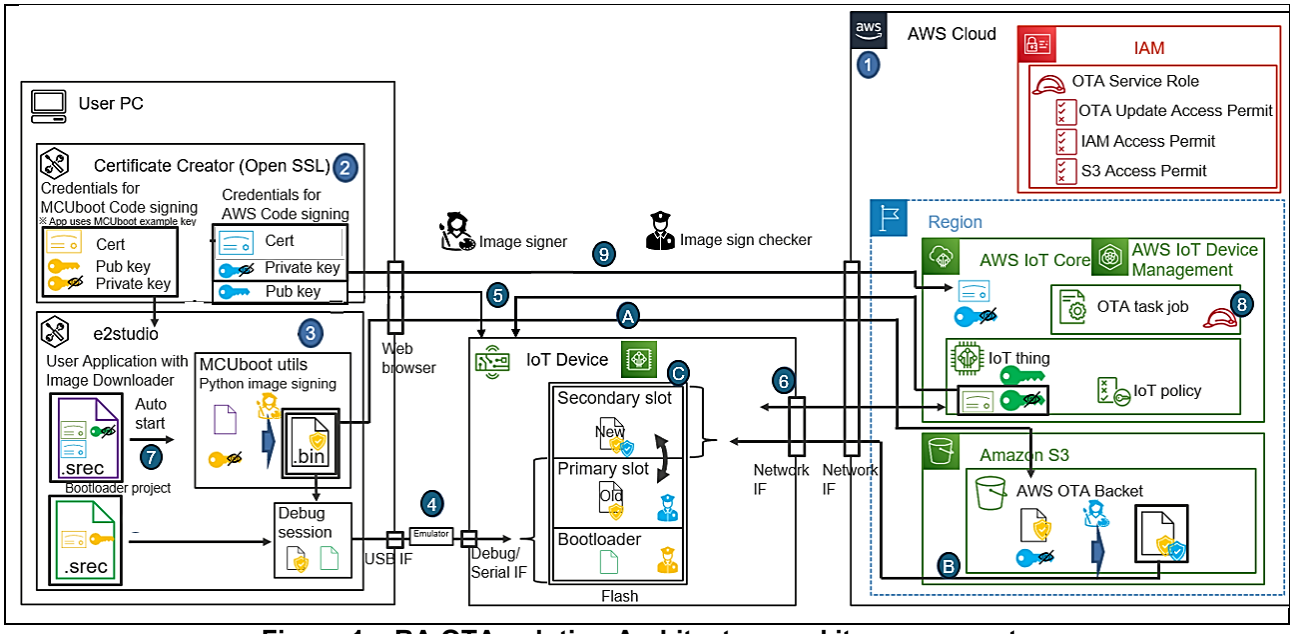

Figure 1. RA OTA solution Architecture and its components

(1): Set up AWS IAM, S3, and IoT Core.

(2): Create credentials using OpenSSL.

(3): Create a bootloader and initialize the user app with a downloader.

(4): Install the bootloader and initialize firmware.

(5): Start application and store Cloud credentials: device credentials, OTA code signing public key.

(6): Communicate with AWS.

(7): Create new firmware to upgrade via OTA.

(8): Create an OTA job.

(9): Install credentials for AWS code signing.

(A): Upload new firmware to S3 Storage.

(B): OTA library download image.

(C): After finishing downloading the image, reset the device, switch, and boot the new image.

Key Components

AWS Cloud side

- AWS IoT Core: Acts as the central messaging hub for IoT devices, facilitating secure and reliable communication between devices and the cloud. Manages device connections, authentication, and message routing.
- AWS S3 Bucket: Primarily, it is used for Firmware Repository. Hosted on Amazon S3 or another suitable storage service within the AWS Cloud. Stores firmware images and associated metadata required for OTA updates.
- Identity and Access Management (IAM): It plays a crucial role in AWS OTA (Over-The-Air) updates by ensuring secure access control and authorization for the various components involved in the update process.

IoT Device

- Secure Bootloader: It ensures the integrity and authenticity of firmware updates by verifying digital signatures and integrity checks, and protects against unauthorized updates and tampering with device firmware.
- Device Firmware Update (DFU) Client (Image Downloader): Software component embedded within IoT devices responsible for managing the OTA update process, communicating with AWS IoT Core, and retrieving firmware updates as directed by AWS IoT Jobs.
- Firmware Image (Application): Represents the software running on IoT devices that may require periodic updates. Contains the application logic, device drivers, and other functionalities specific to the device's operation.

System Setup and Image Creation

- Certificate Generation: Certification Creation using OpenSSL includes OpenSSL, which serves as the primary tool for managing digital certificates and cryptographic keys. The process typically begins with the generation of a private key, a cryptographic key used for generating digital signatures and encrypting data. A Certificate Signing Request (CSR) is then created, containing information about the entity requesting the certificate and its public key. This CSR is submitted to a Certificate Authority (CA), responsible for issuing digital certificates. The resulting certificate binds the entity's public key to its identity and authenticity, providing proof of its identity. Throughout this process, OpenSSL facilitates the generation of keys, CSRs, and certificates, as well as the interaction with the CA.
- Image Signing: AWS OTA involves digitally signing firmware images before deployment to IoT devices. This process ensures the authenticity and integrity of the firmware updates, mitigating the risk of unauthorized modifications or tampering. By using cryptographic signatures, such as RSA or ECDSA, firmware images are signed with a private key held by the firmware provider. Devices receiving the update can verify the signature using the corresponding public key, thereby confirming the authenticity of the firmware before installation. Image signing is a critical security measure in OTA updates, enhancing trust and reliability in the update process for IoT devices.
- Creation of Bootloader Image: Involves developing a bootloader software component that ensures the integrity and security of firmware updates on IoT devices. A secure bootloader verifies the authenticity of firmware images before allowing them to be installed on the device. This process typically involves digitally signing the bootloader and firmware images using cryptographic signatures. The bootloader verifies the digital signatures of the firmware images against a trusted public key, ensuring that only authorized and unaltered firmware updates are installed. By incorporating a secure bootloader into the OTA update process, IoT device manufacturers can enhance security, protect against unauthorized modifications, and maintain the integrity of their device's firmware.

- Creation of Application Firmware Image: Encompasses the development of device application firmware that includes both the device firmware upgrade (DFU) functionality and the downloader agent. This comprehensive firmware image enables seamless over-the-air (OTA) updates for IoT devices. The DFU functionality allows devices to receive and apply firmware updates securely, ensuring compatibility, stability, and enhanced features. Concurrently, the downloader agent facilitates the retrieval of firmware updates from remote servers or cloud platforms, managing the download process efficiently.

1.3 Preparing for OTA Update using Renesas RA MCU

Creating an OTA update using Renesas RA MCU involves many logical steps. In this section, the details of the steps and creations are explained.

1.3.1 Creating a Bootloader Project for the Device

Creating a bootloader project for a device involves developing a specialized software component responsible for initializing the device and loading the main application firmware. The bootloader is typically the first code executed when the device powers on and plays a crucial role in the system's boot process. When the OTA of the new application firmware is completed and written to the designated flash area, for the new application firmware to work, the bootloader program switches to the new firmware image and boots the new firmware using RA MCUboot by resetting the MCU.

This Bundle includes a Bootloader Project to work with the associated Application Project. The **“R11AN0915”** document contains the section **“4.3 Creating the Application Project using the FSP Configurator”**, where the creation of the Bootloader Project is explained to the users.

1.3.2 Generating SSL certificates for the secure Image

Generating SSL certificates for secure images involves creating digital certificates to establish secure communication channels between devices and servers. These certificates play a crucial role in ensuring data confidentiality, integrity, and authentication in IoT applications. More details on creating certificates and Key pairs, adding them to the bootloader project, and creating a Python signing environment are explained in the section **3.2.**

1.3.3 Creating an Application Project with the Downloader

This Bundle includes an Application Project to work with the associated App note. This Application Project includes a downloader and OTA agent to work with the AWS OTA Manager. “**R11AN0915**” document contains section **“4.3 Creating the Application Project using the FSP Configurator”**, where step-by-step creation of the Application Project is explained to the users.

As a general use case, for the RA OTA solution of the IoT device, the firmware will be deployed to multiple devices. Therefore, the firmware image should not contain device-specific information like device credentials. In this demo project, the following information can be set at runtime.

- AWS IoT Thing name.
- AWS IoT Device credential – Certificate.
- AWS IoT Device credential – Private Key.
- AWS Code Signing.
- Ethernet MAC address.

1.3.4 Creating FSP Solution Project to link Bootloader and Downloader projects

Starting with FSP v6.0.0 onwards, you must create an FSP Solution project to configure bootloader slots, slot sizes, and header information, which were previously managed within the MCUboot module. The document ‘**R11AN0915**’ includes section “**4.4 Creating the FSP Solution Project**”, which provides step-by-step instructions for creating the solution project and configuring the flash layout memory

1.3.5 Establishing a connection to the AWS IoT Core

The Application project must establish a connection to AWS IoT Core in the cloud, utilizing connectivity protocols like HTTP or MQTT. This App note specifically demonstrates the OTA feature using the MQTT

protocol. AWS IoT establishes a secure connection to the IoT device through MQTT/TLS. The MQTT Client, integrated with FSP, enables users to connect and initiate communication with the Cloud. Initially, on the device side, authentication of the server and configuration of the IoT endpoint are required for connection, along with the utilization of device credentials generated by AWS IoT Core. Further elaboration on this aspect is provided in section 4 Running the Application Project.

1.3.6 Setting up the Cloud for the OTA

AWS OTA updates require an AWS IAM or root user account. AWS IoT Core will be used to authenticate access to AWS IoT resources. An Amazon S3 bucket is used to store firmware images or files that need to be updated on the device. Sections 2 and 5.2 describe the details of setting up AWS resources for OTA firmware updates.

2. Configuring AWS Cloud

To run the RA OTA solution using the FreeRTOS demo, you must have an AWS account (the root user or an IAM user with permissions to access AWS IoT and FreeRTOS cloud services).

For details on how to sign up for an AWS account and add permissions to users, see

https://docs.aws.amazon.com/freertos/latest/userguide/freertos-prereqs.html.

For details on how to set up OTA updates, see https://docs.aws.amazon.com/freertos/latest/userguide/ota-prereqs.html.

You must then register the board with AWS IoT by following the instructions in https://docs.aws.amazon.com/freertos/latest/userguide/freertos-prereqs.html#get-started-freertos-thing.

Note: The Snapshots from AWS shown here are the current state at the time of writing the document. Some of the snapshots may look different. If the displayed content is different, check the documents provided by AWS.

2.1.1 Signing in to the AWS Console

(1) Access the AWS website (https://aws.amazon.com/) and click **Sign In**

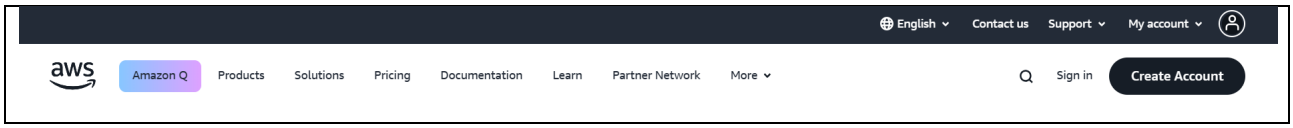

Figure 2. Sign In

(2) Enter your email address or account ID, and then click **Next**

If the account you are using to sign in is the root user, enter the root user's email address. If the account is an IAM user, enter the account ID. (You might skip this step if you have already signed in)

Figure 3. Enter Root user email address

(3) Enter your password and then click **Sign in**

- For root users

Figure 4. Enter your password (For root users)

- For IAM users

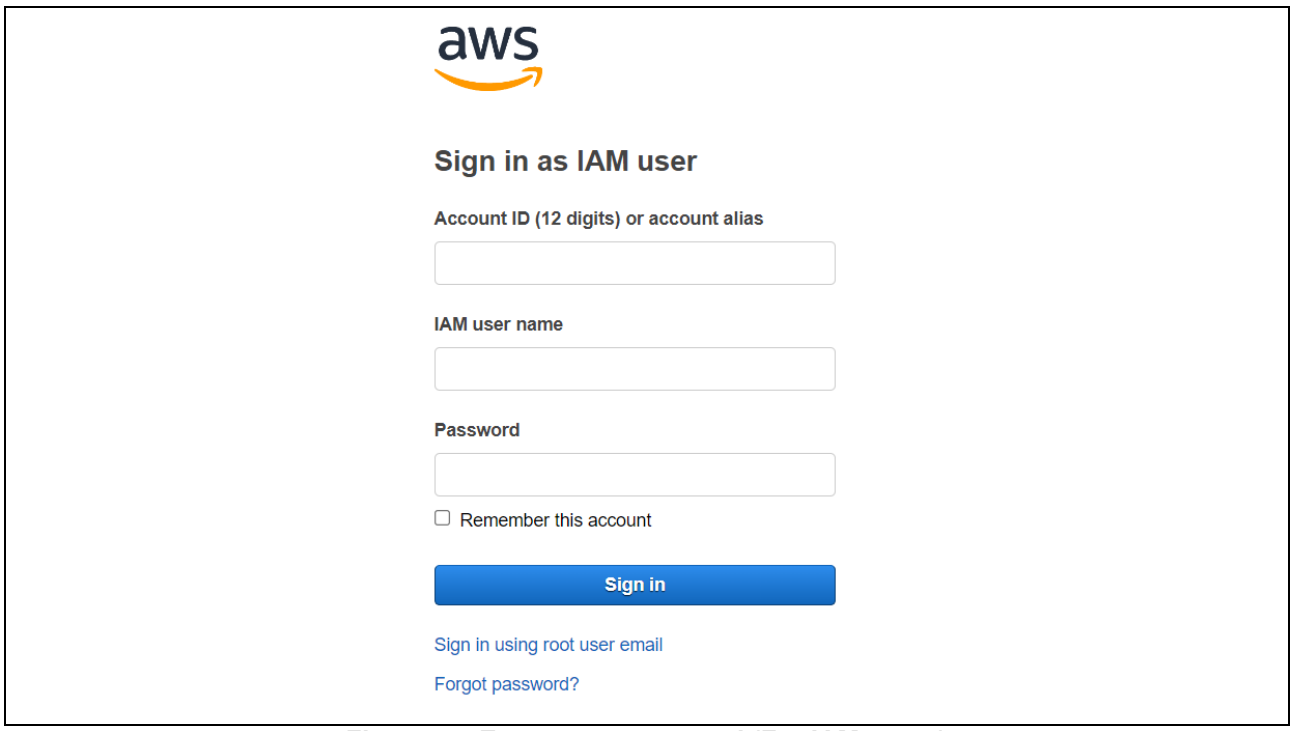

Figure 5. Enter your password (For IAM users)

2.1.2 Setting your region in AWS

After logging in to AWS, select your region in the top right of the screen.

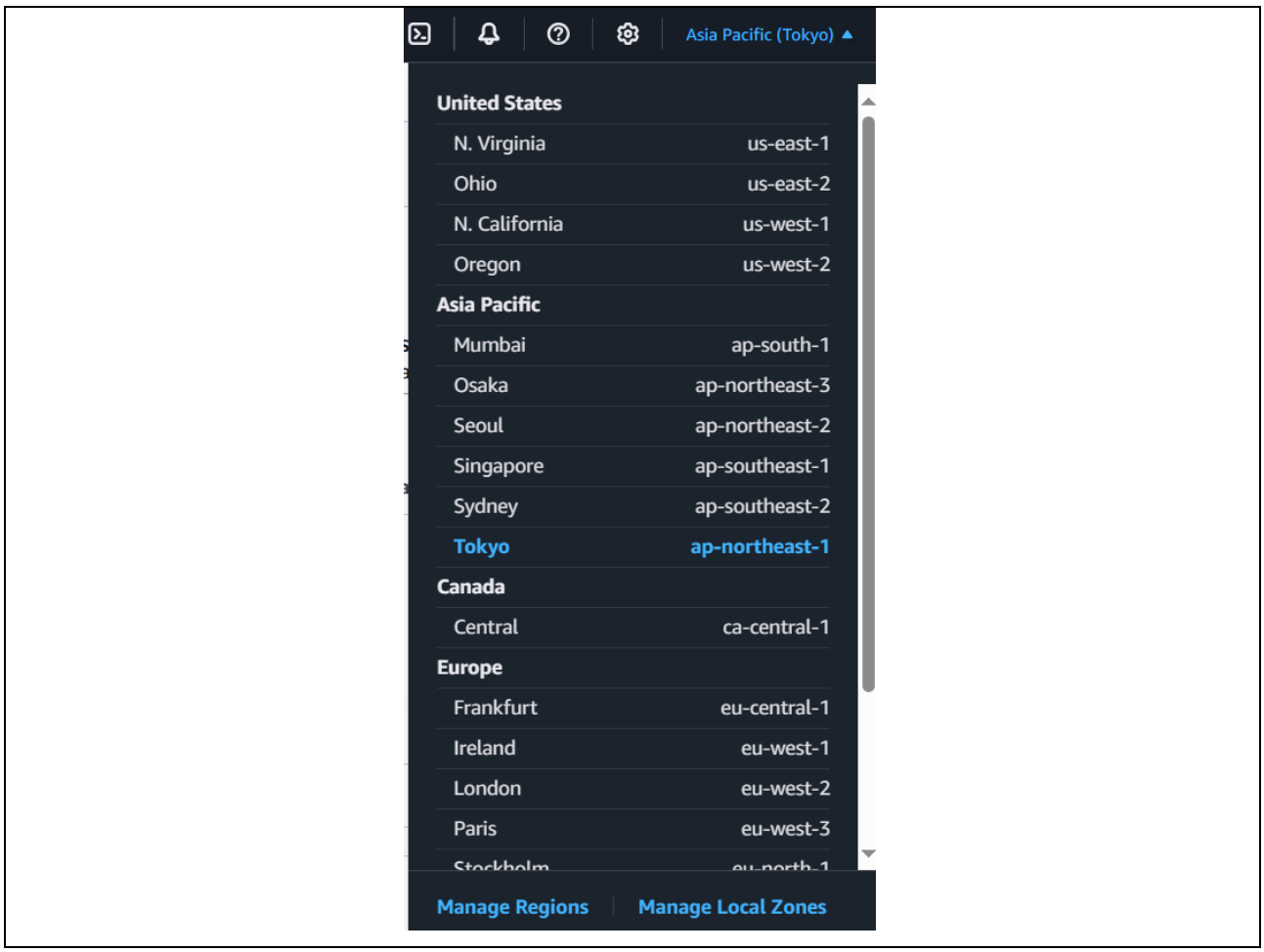

Figure 6. Select your region

2.1.3 Registering your device in AWS

The following explains the preparations necessary to run the demo project in AWS. Set up AWS by referring to the following tutorial.

2.1.3.1 Setting policies

Assign access permissions (policies) for AWS and other resources to the device you want to connect to AWS.

Assign the following policies to the device connected in this application note:

- **iot:Connect**: Connects to AWS IoT
- **iot:Publish**: Publishes a topic
- **iot:Subscribe**: Subscribes to a topic
- **iot:Receive**: Receives messages from AWS IoT

(1) Enter IoT Core in the search box at the top of the screen, and click **IoT Core** in the search results

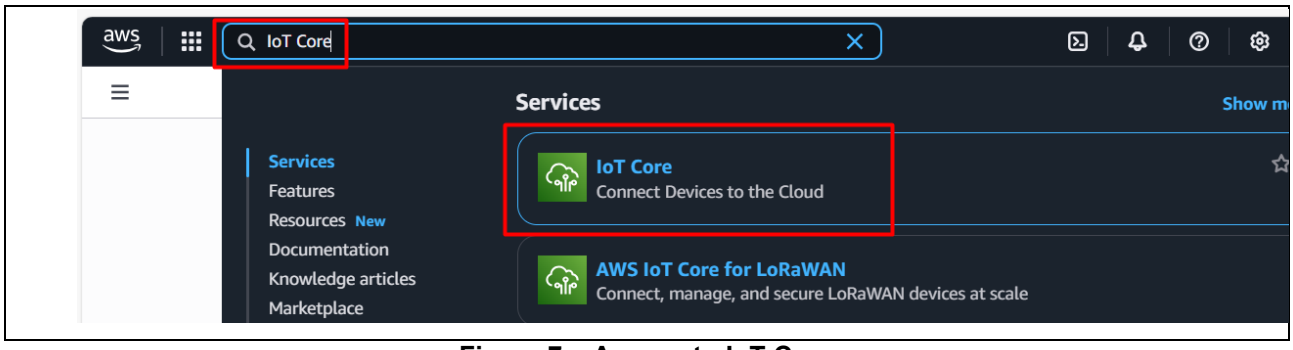

Figure 7. Access to IoT Core

(2) In the menu, click **Security** and then **Policies**, and then click the **Create policy** button

Figure 8. Click the Create policy button

(3) Enter a policy name (for example: ra6m5_ota_demo_policy)

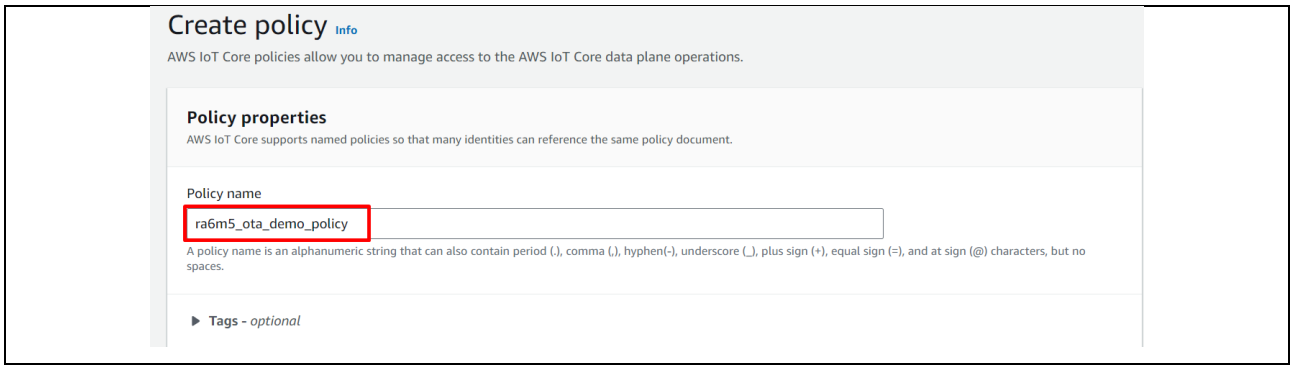

Figure 9. Create a policy name

(4) Click the **Policy statements** tab, and in the **Policy document** area, click **Builder**. Enter the policy settings as shown in the following figure, and then click **Create**

Because the policy initially contains only one statement, you must add more statements by clicking the **Add new statement** button, then choosing **Policy action**, **Policy effect,** and **Policy resource** (fill in the value **"*"** in the blank) as shown below:

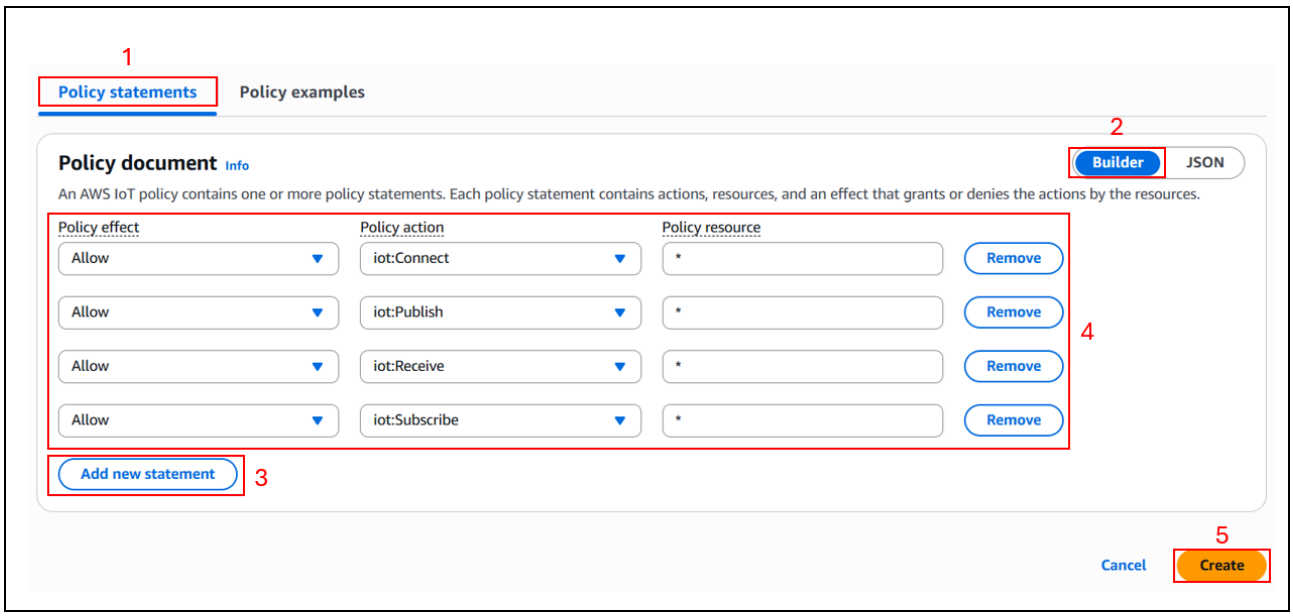

Figure 10. Create policy statements

2.1.3.2 Registering your device as a thing in AWS IoT

(1) In the menu, click **Manage**, **All devices**, and **Things**, and then click the **Create things** button

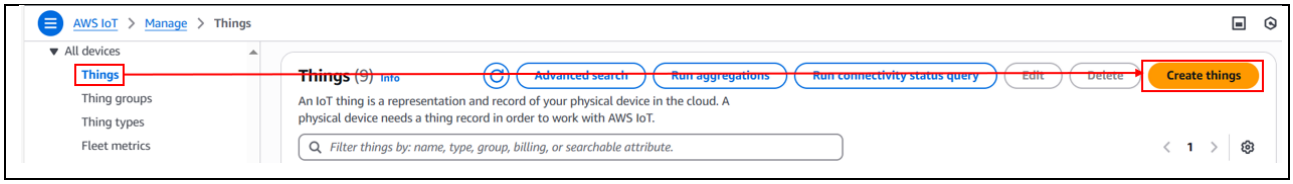

Figure 11. Click the Create things button

(2) Select **Create single thing** and then click **Next**

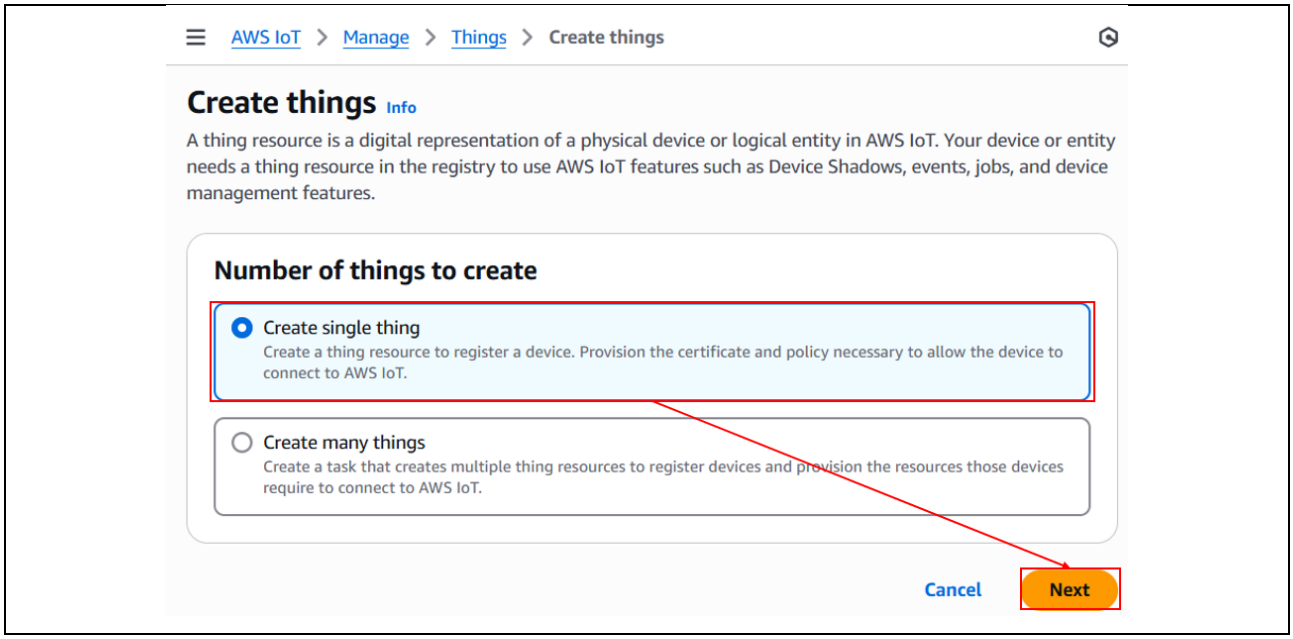

Figure 12. Choose the Create single thing

(3) Enter a thing name (example: ra6m5_ota_demo_thing), and then click **Next**

Take note of the name of the thing you entered. You will need it later.

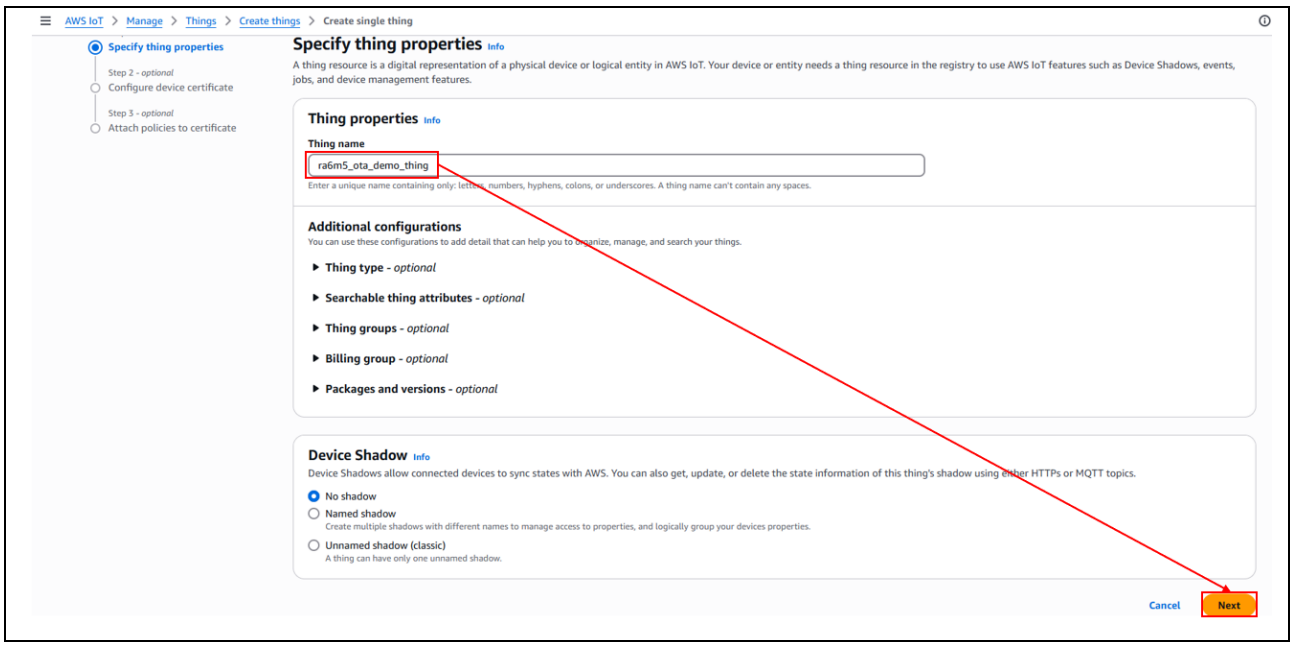

Figure 13. Configure the Specify thing properties

(4) In the Device certificate area, select **Auto-generate a new certificate** and then click **Next**

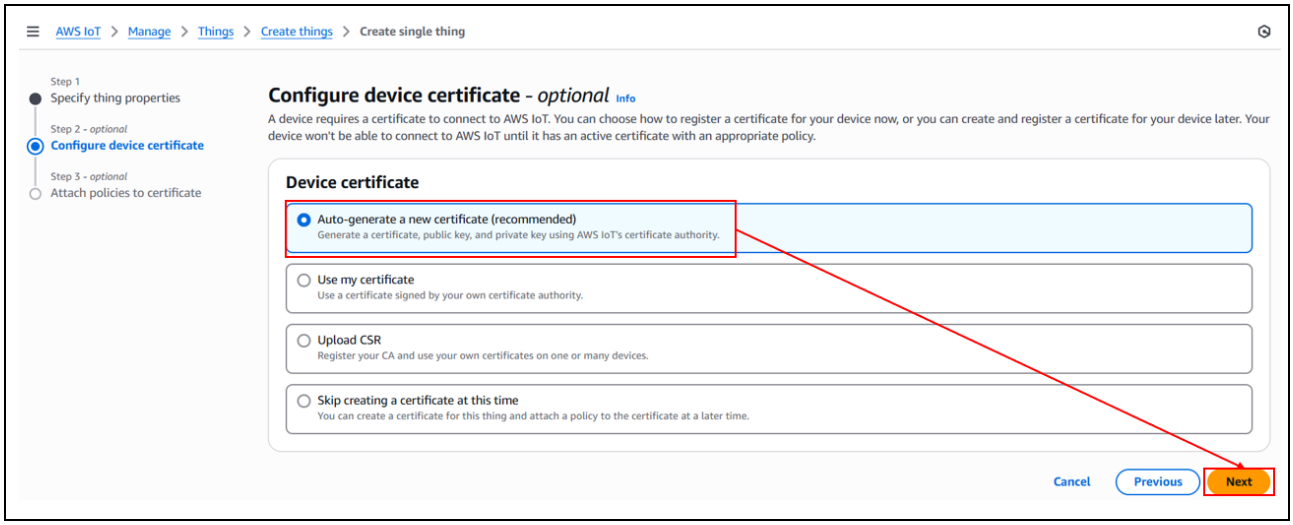

Figure 14. Configure device certificate

(5) Attach the policy to the certificate

Select the policy you created in section **2.1.3.1**, and then click the **Create thing** button

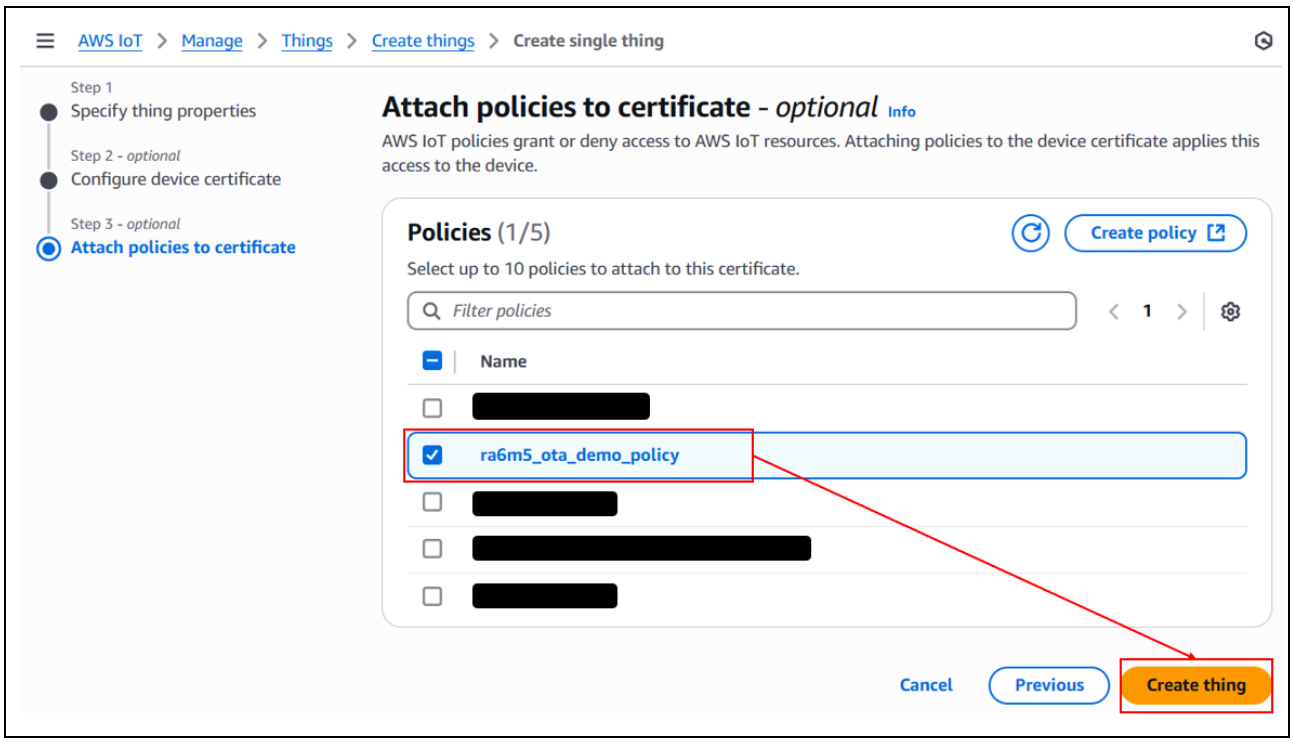

Figure 15. Attach policies to the certificate

(6) Download the certificate and key files

The certificate and private key are equivalent to a password for the device (thing). When you register a certificate and private key on a device, the device can use this certificate and private key to connect to AWS.

You must download the certificate, public key, and private key now. You will not have another opportunity to download them.

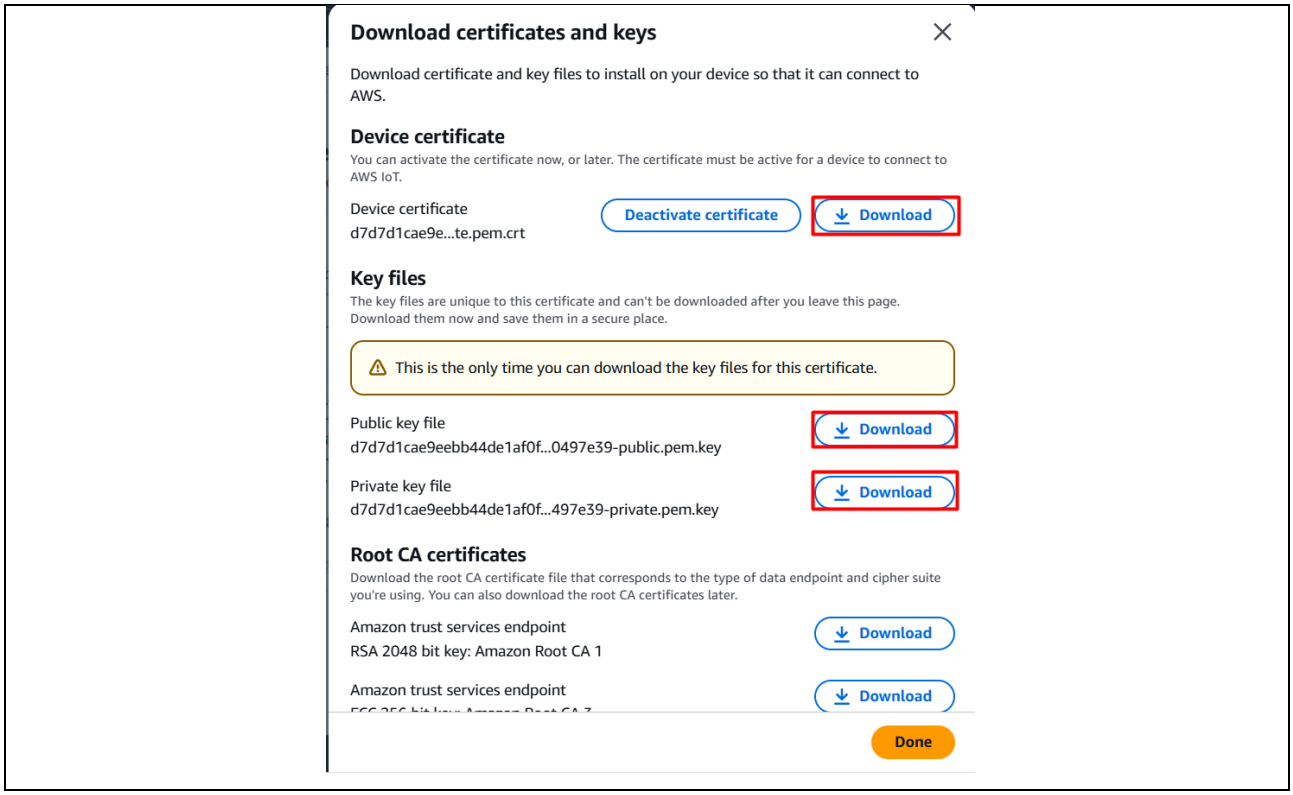

Figure 16. Download the certificate and key files

2.1.3.3 Checking the endpoint

The endpoint is equivalent to a connection destination (URL) for the device (thing). The device will connect to the endpoint registered for the device.

In the menu, click Connect, and then click **Domain configurations,** and make a note of the endpoint (copy endpoint site).

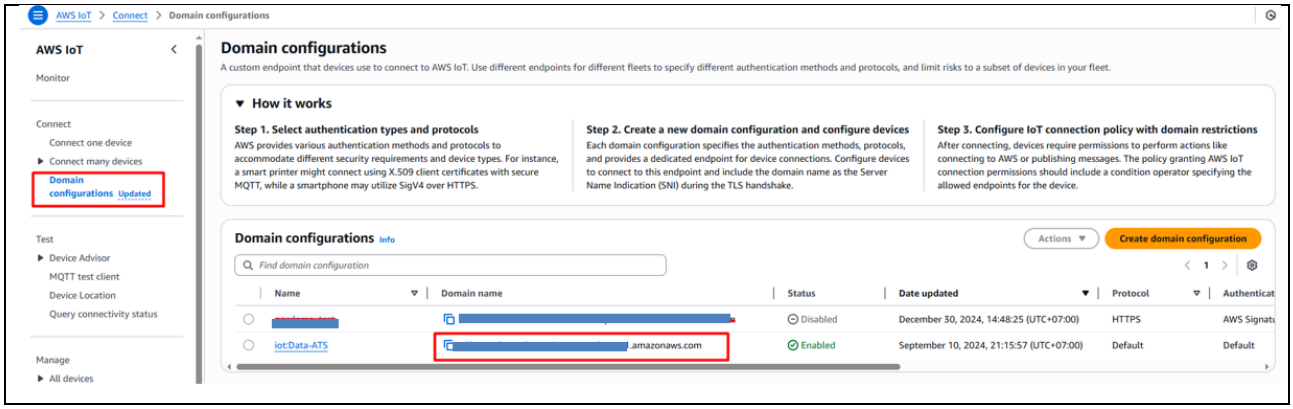

Figure 17. Make a note of the endpoint

2.1.3.4 Creating an Amazon S3 Bucket

Amazon S3 is an online storage web service that stores the firmware with which the device will be updated.

(1) From the **Services** menu, select **Storage** and then **S3**

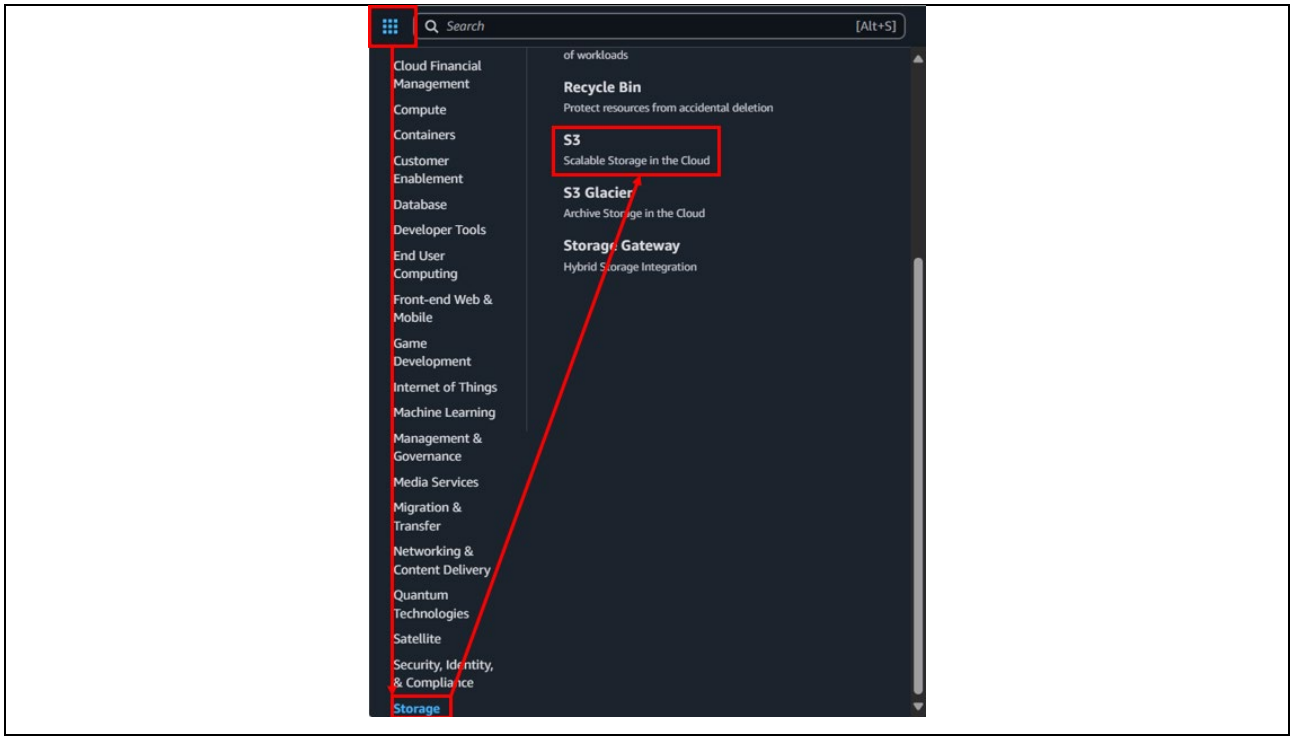

Figure 18. Select S3 in the Storage

(2) On the **General purpose buckets** page, click the **Create bucket** button

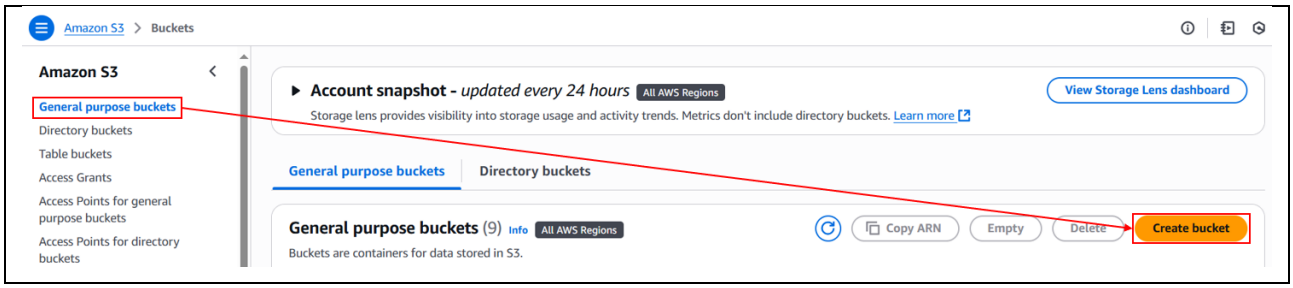

Figure 19. Click the Create bucket button

(3) Enter a bucket name (example: s3test-ra6m5v2)

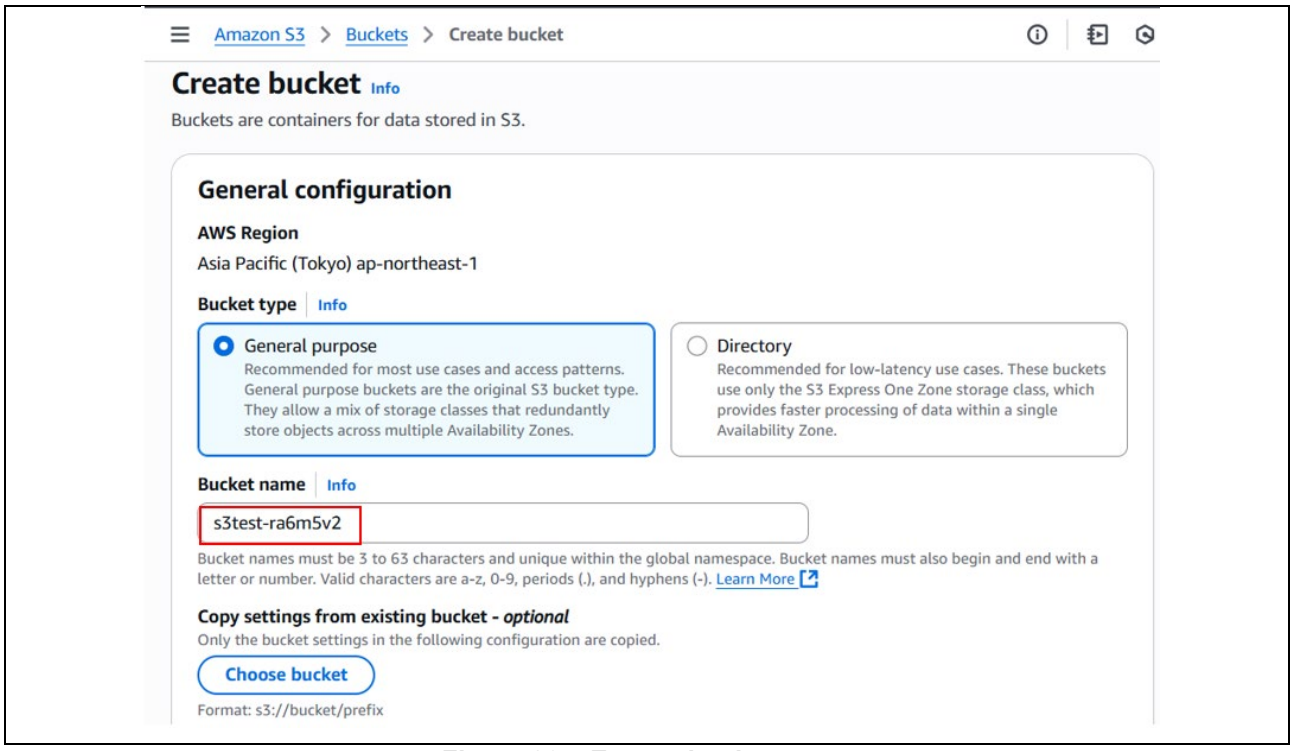

Figure 20. Enter a bucket name

The bucket name must be unique in your account. The following error message appears if the bucket name is already in use. In this case, use another name.

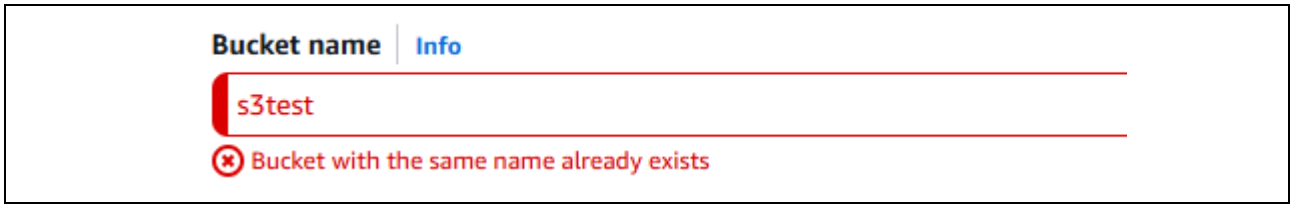

Figure 21. Error message if the bucket name is already in use

(4) Create the bucket

Enter the settings as follows, and then click the **Create bucket** button

- **Block Public Access setting for this bucket**: **Block all public access**
- **Bucket Versioning**: **Enable**

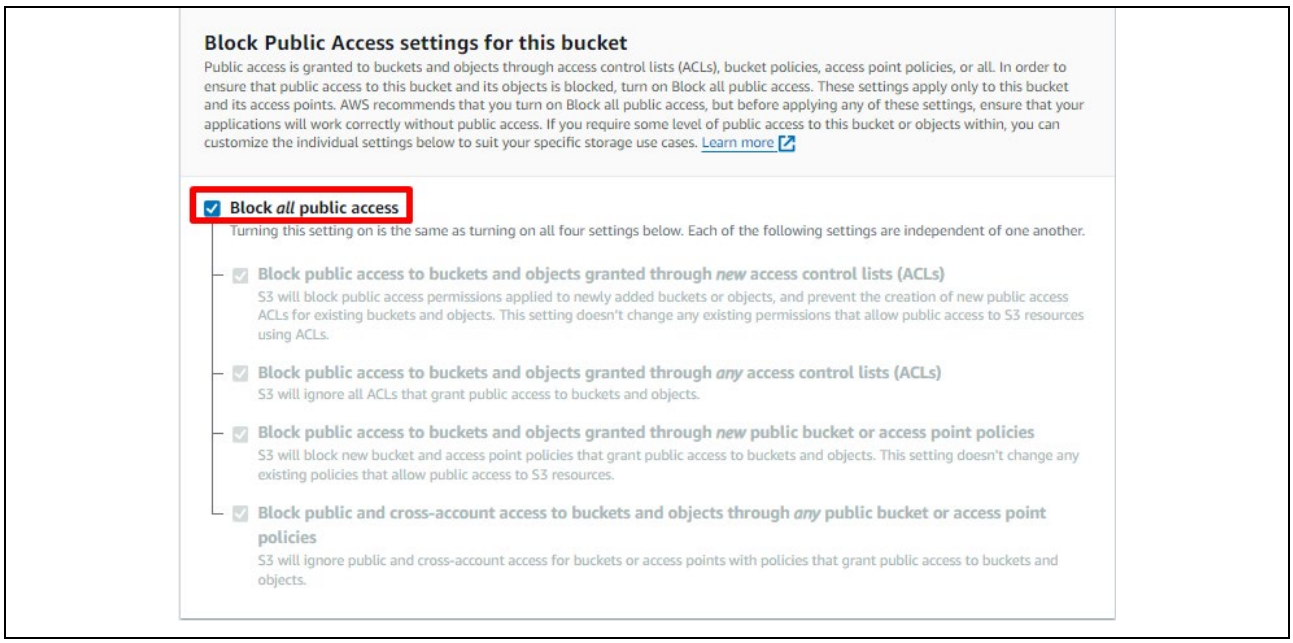

Figure 22. Choose Block all public access

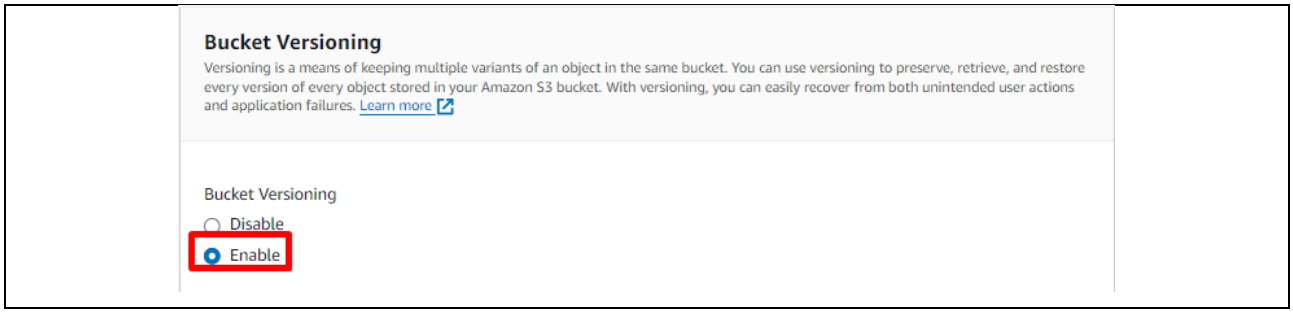

Figure 23. Enable Bucket Versioning

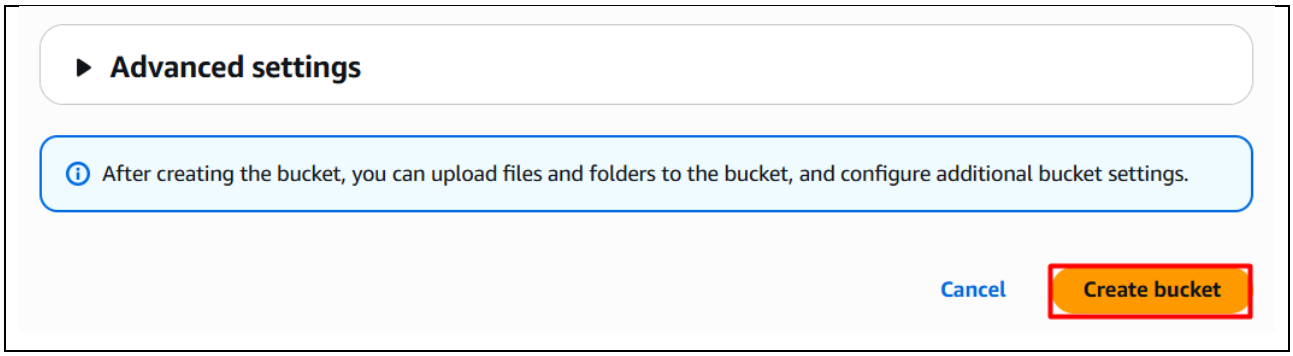

Figure 24. Choose the Create bucket button

2.1.3.5 Allocating OTA execution permission to IAM users

Create a role with the appropriate access permissions to create OTA update jobs.

(1) Enter IAM in the search box at the top of the screen, and click **IAM** in the search results

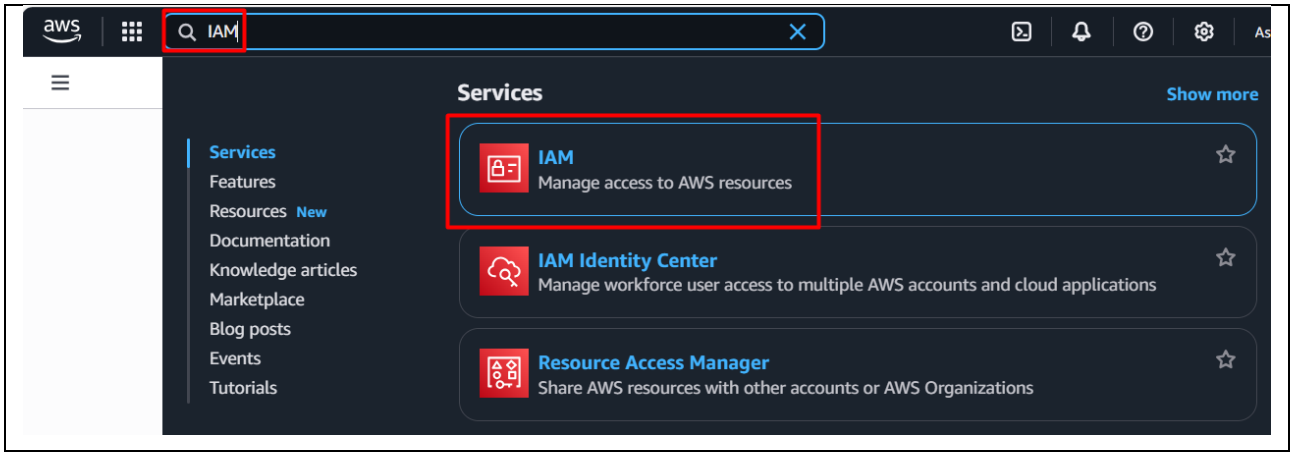

Figure 25. Choose IAM service

(2) In the menu, click **Roles** and then click the **Create role** button

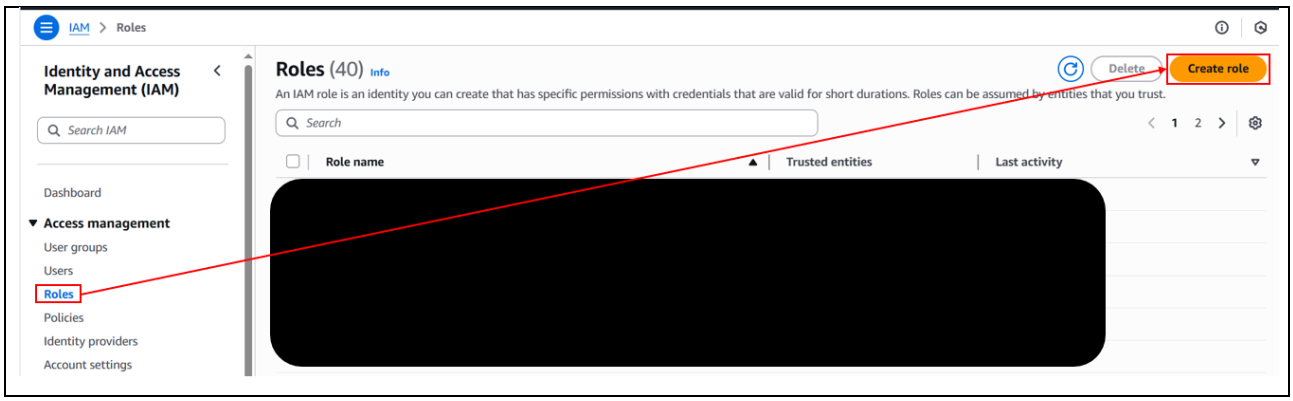

Figure 26. Click the Create role button

(3) Under **Select trusted entity**, enter the following settings, and then click **Next**:

- Under **Trusted entity type**, select **AWS service**
- Under **Use case for other AWS services**, enter IoT in the Service or use case search box and press the Enter key.
- Select the **IoT** option button

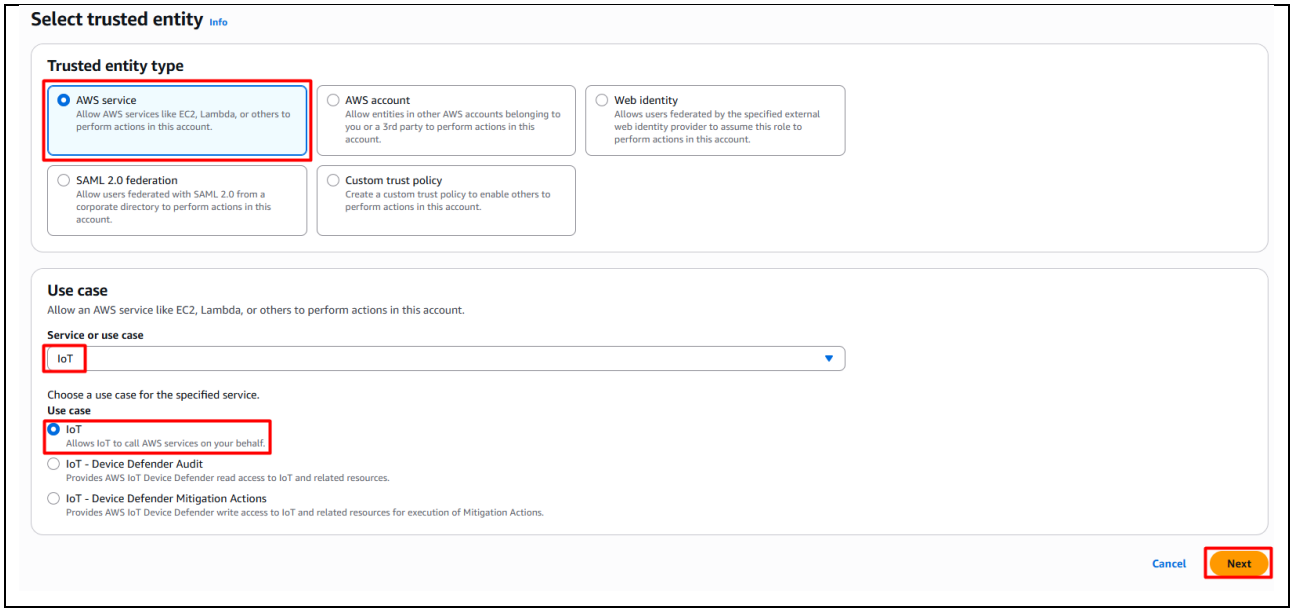

Figure 27. Configure the Select trusted entity

(4) Click **Next** on the **Add permissions** page without making any changes

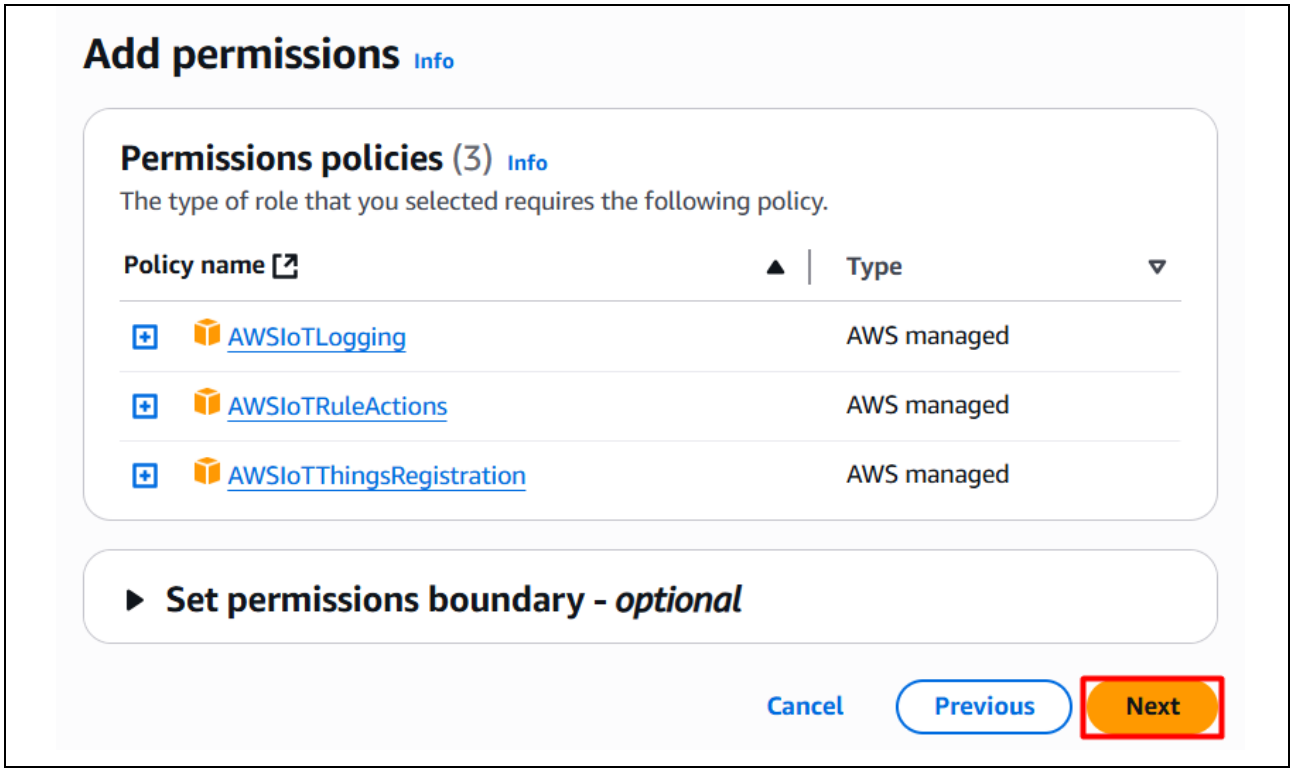

Figure 28. Add permissions

(5) Enter a role name (example: ota_role_ra6m5v2), and then click the **Create role** button

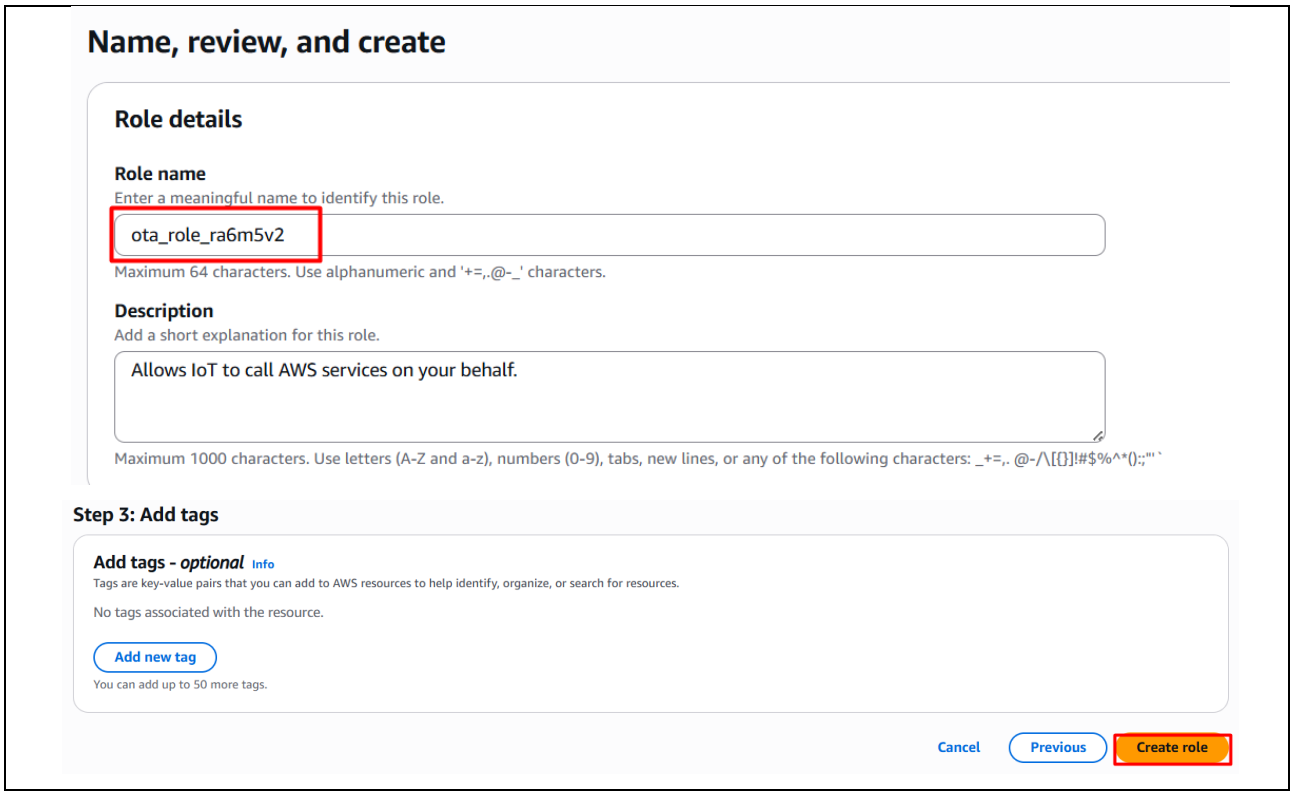

Figure 29. Enter a role name

(6) Click the role you created

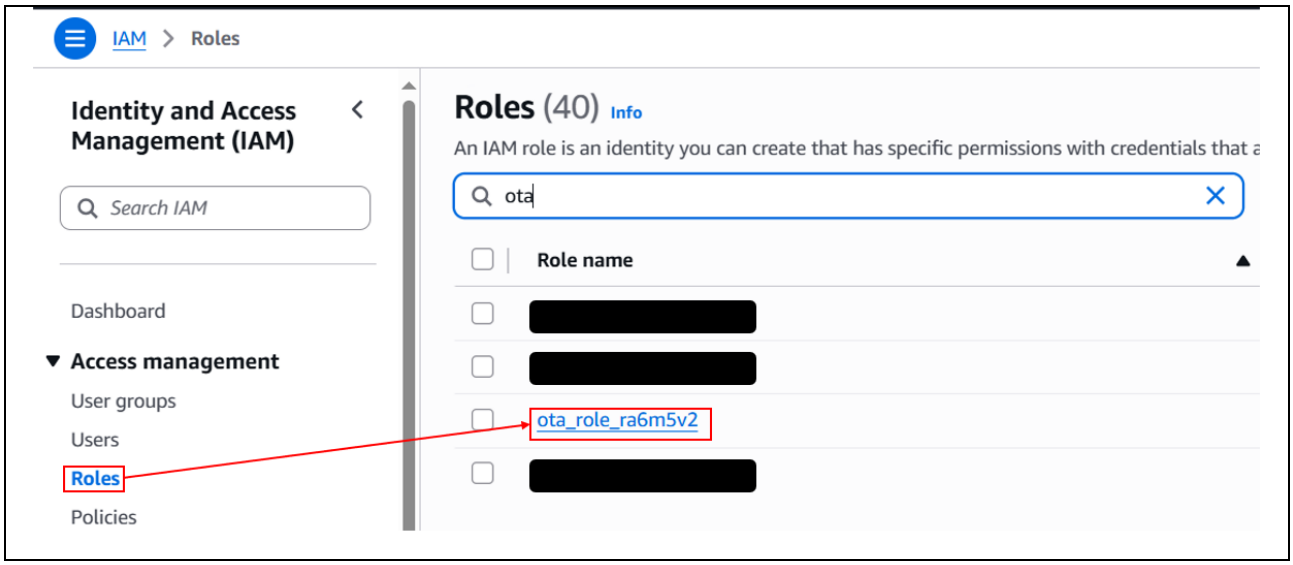

Figure 30. Click the role you created

(7) From the **Add permissions** drop-down list, select **Attach policies**

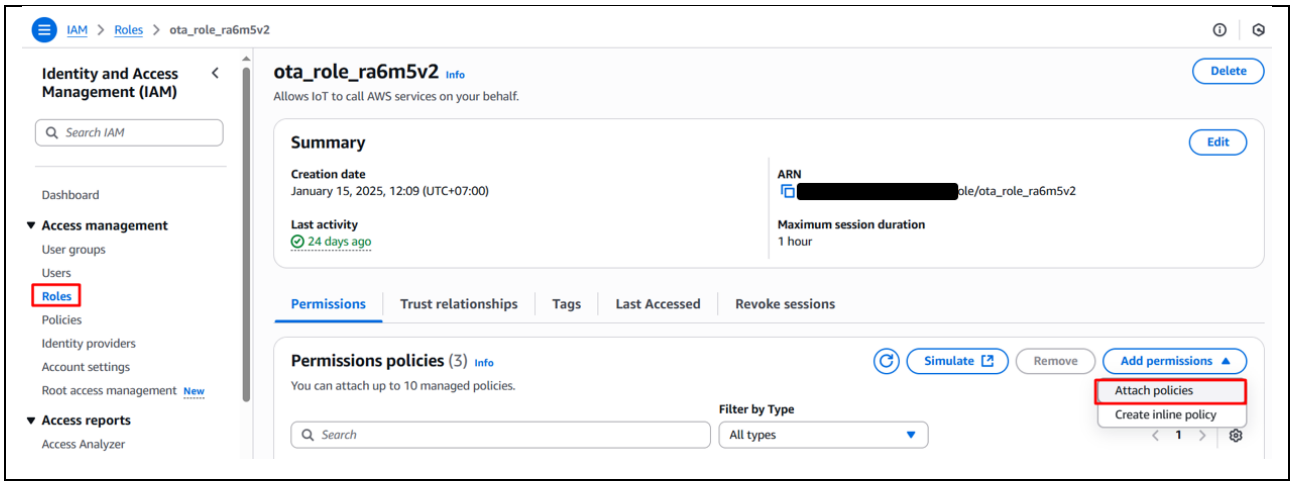

Figure 31. Select the Attach policies

(8) Enter AmazonFreeRTOSOTAUpdate in the **Permissions policies** search box

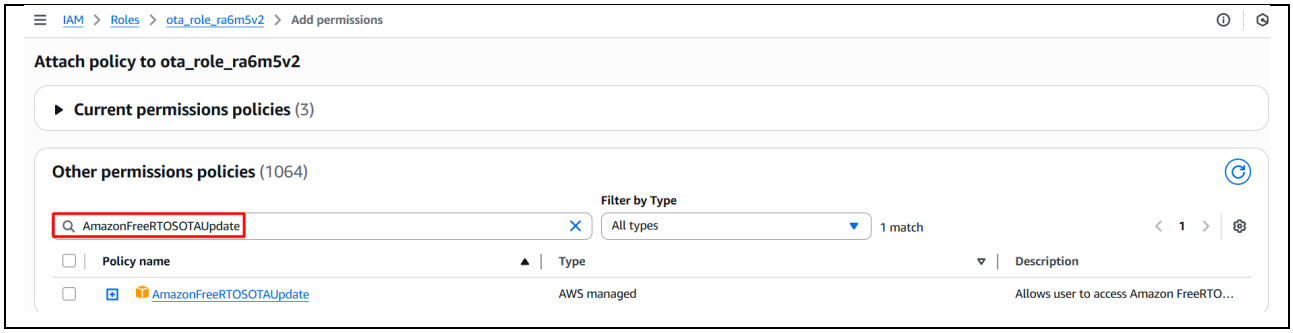

Figure 32. Search for the AmazonFreeRTOSOTAUpdate policy in the Permissions policies

(9) Select the check box beside the AmazonFreeRTOSOTAUpdate policy, and then click the **Add permissions** button

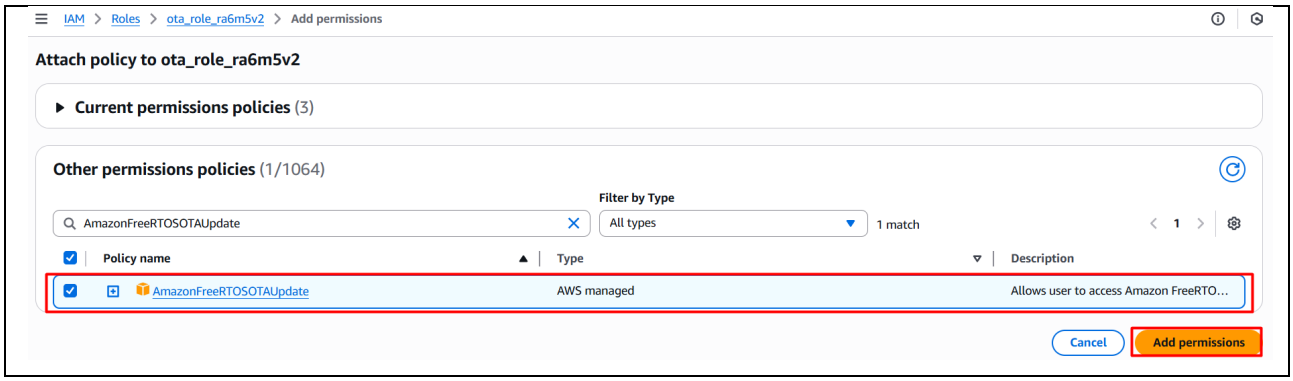

Figure 33. Add AmazonFreeRTOSOTAUpdate policy

(10) From the **Add permissions** drop-down list, select **Create inline policy**

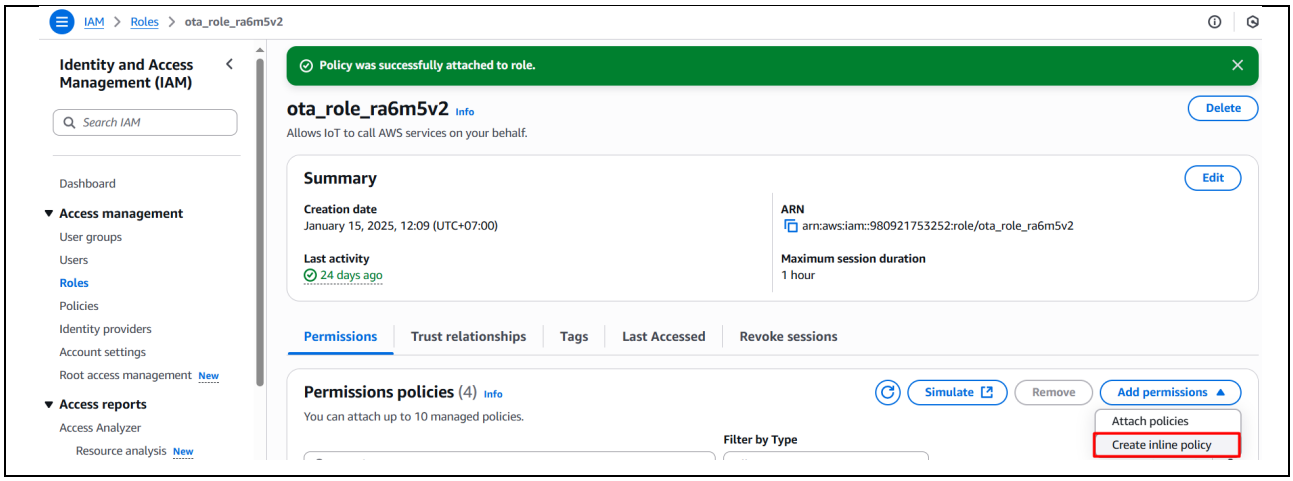

Figure 34. Select the Create inline policy

(11) Click **JSON**, paste the following code, and then click **Next**

This code grants permission to pass the IAM role to AWS services.

Code to paste:

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

Figure 35. Add Specify permissions

(12) Enter a policy name (example: ra6m5_ota_demo_iam_policy), and then click the **Create policy** button

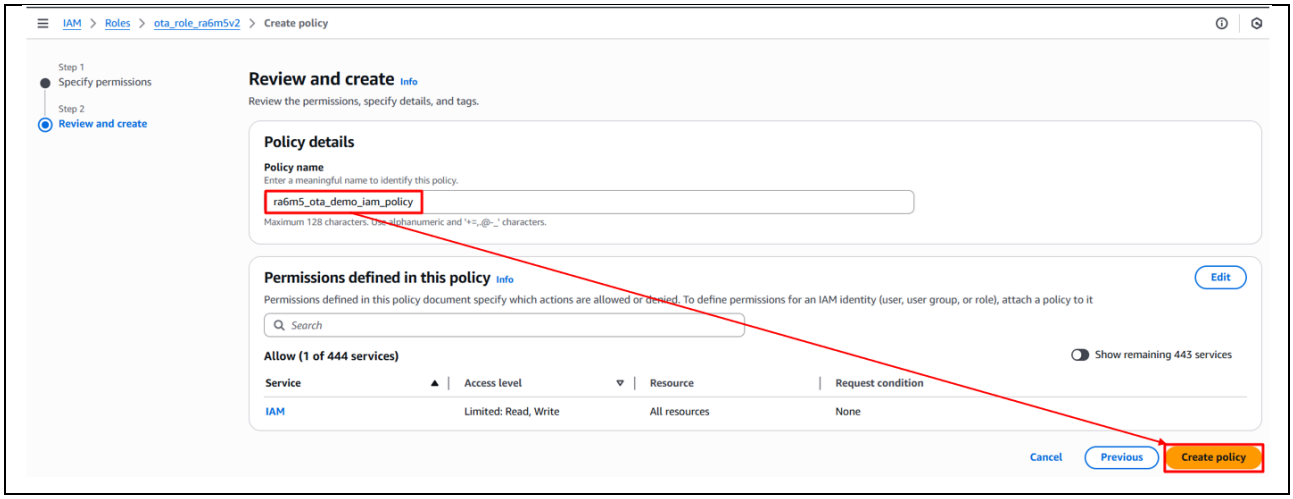

Figure 36. Enter a policy name

(13) Again, from the **Add permissions** drop-down list, select **Create inline policy**

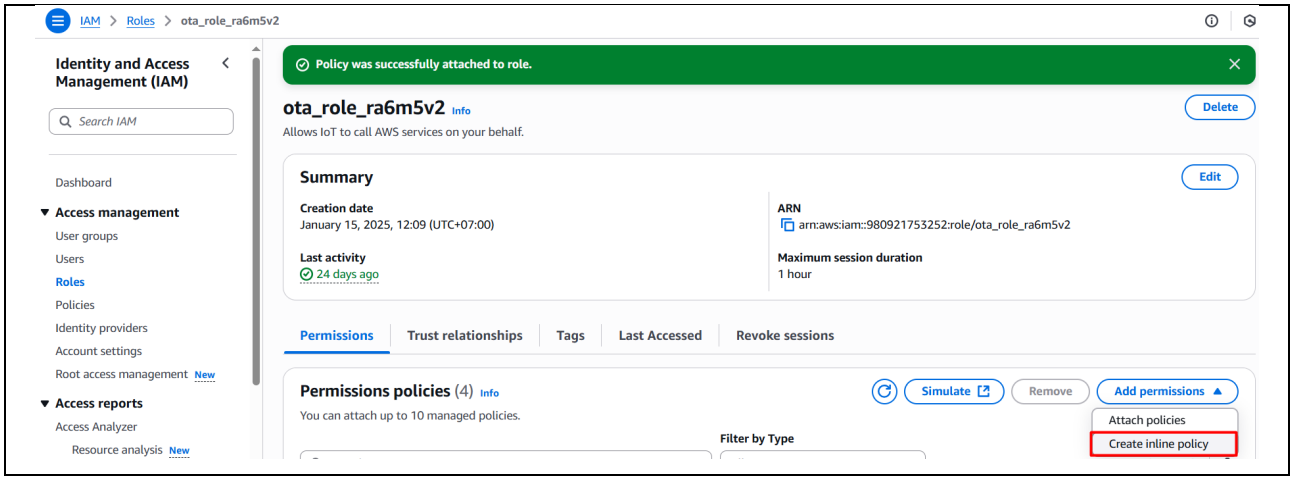

Figure 37. Select the Create inline policy

(14) Click **JSON**, paste the following code, and then click **Next**

This code allows access to Amazon S3, where the updated firmware is stored.

Code to paste:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "s3:GetObjectVersion",
            "s3:GetObject",
            "s3:PutObject"
        ],
        "Resource": [
            "*"
        ]
    }
  ]
}
```

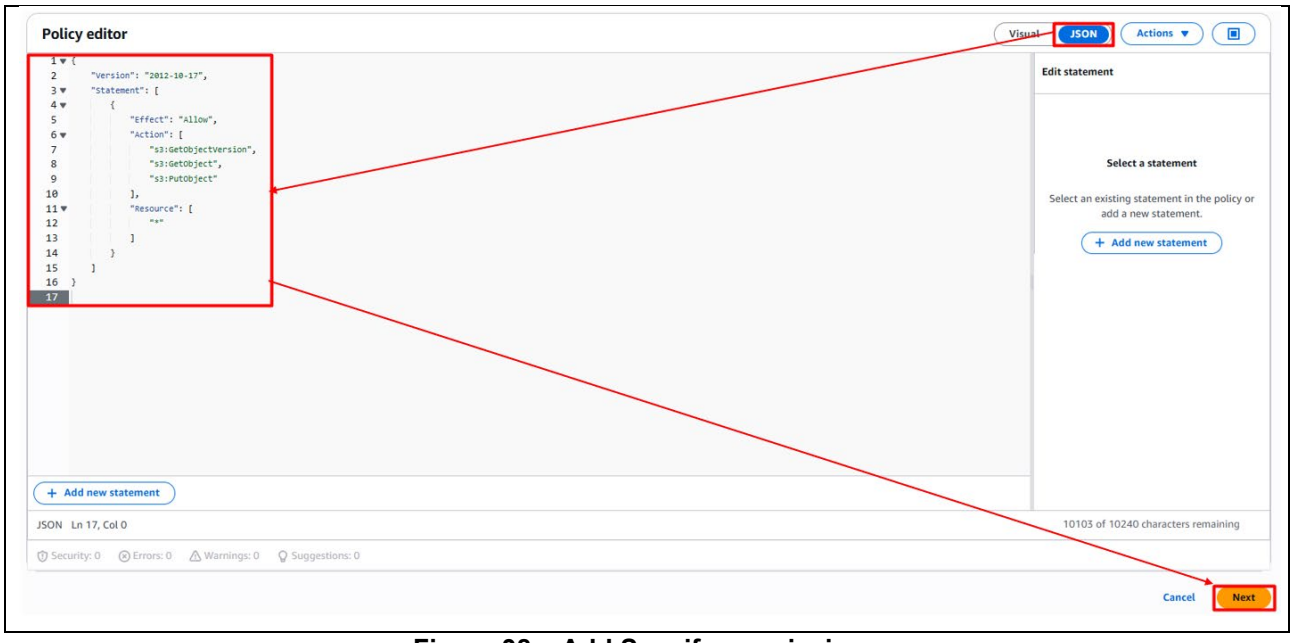

Figure 38. Add Specify permissions

(15) Enter a policy name (example: ra6m5_ota_demo_s3_policy), and then click the **Create policy** button

Figure 39. Enter a policy name

3. Importing, Setting, Building, and Loading the Project

Note: The Application Project bundled as part of the App note can be imported, and necessary changes to the settings can be made to work with a few modifications. If the user wants to create a Bootloader and Application Project from scratch, please refer to the App Note “R11AN0915”, where step-by-step creation of the Projects is explained.

3.1 Importing

Project “`aws_ck_ra6m5_v2_ethernet_ota_solution`”, project “`aws_ck_ra6m5_v2_ethernet_ota_app`”, and project “`bootloader_ck_ra6m5`” can be imported into the e² studio using the instructions provided in the RA *FSP User’s Manual*. See section *Starting Development > e² studio User Guide > Importing an Existing Project into e² studio.*

3.2 Setting Up the Device

3.2.1 Generating key pairs and certificates

(1) From the Windows OS Start menu, open the Win64 OpenSSL Command Prompt

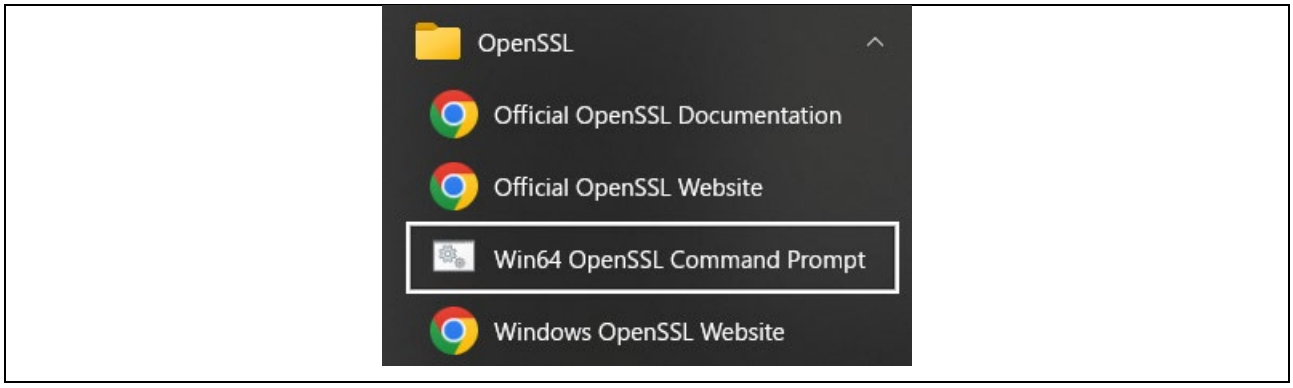

Figure 40. Open the Win64 OpenSSL Command Prompt

(2) Execute the command to create a CA private key using ECDSA

Execute the following command:

“`openssl ecparam -genkey -name secp256r1 -out ca.key`”

Execution results:

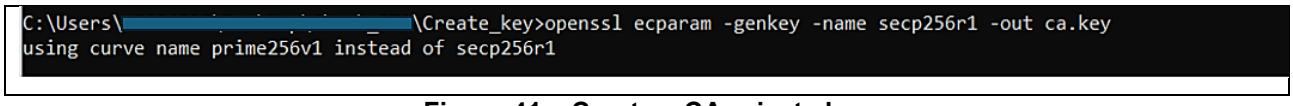

Figure 41. Create a CA private key

(3) Execute the command to create a CA certificate from the CA private key you created.

Execute the following command: You can enter any character string for **Country Name** onward.

"openssl req -x509 -sha256 -new -nodes -key ca.key -days 3650 -out ca.crt"

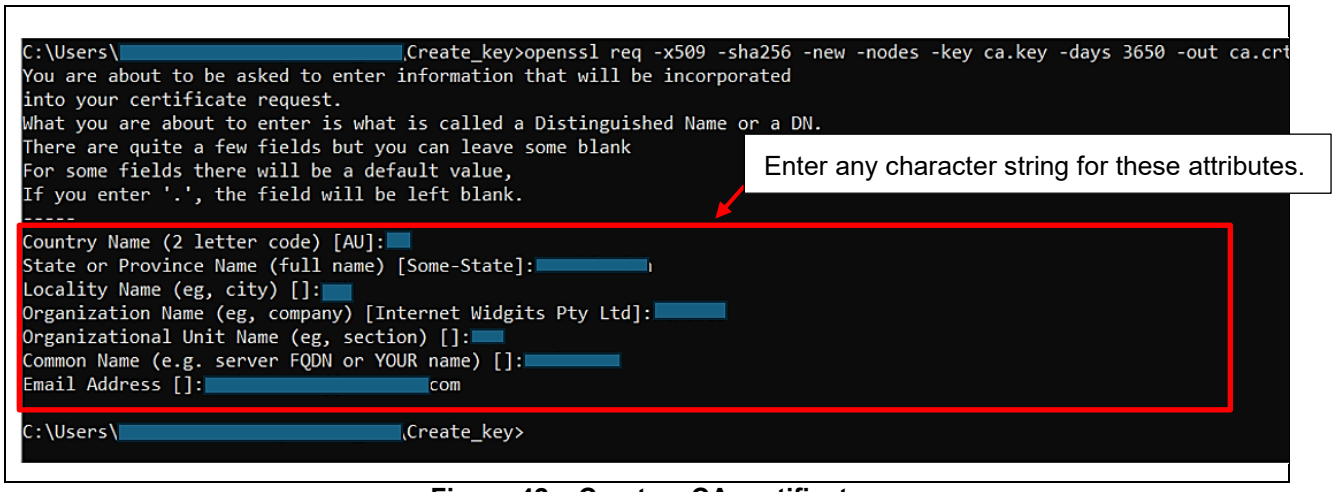

Figure 42. Create a CA certificate

(4) Execute the command to create an ECDSA key pair

Execute the following command:

"openssl ecparam -genkey -name secp256r1 -out secp256r1.keypair"

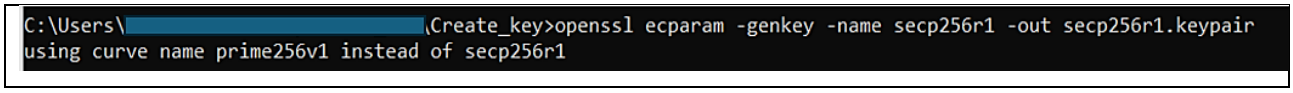

Figure 43. Create an ECDSA key pair

(5) Execute the command to create a certificate signing request from the ECDSA key pair you created

Execute the following command: You can enter any character string for **Country Name** onward. For the last two lines, press **Enter** without entering anything.

"openssl req -new -sha256 -key secp256r1.keypair > secp256r1.csr"

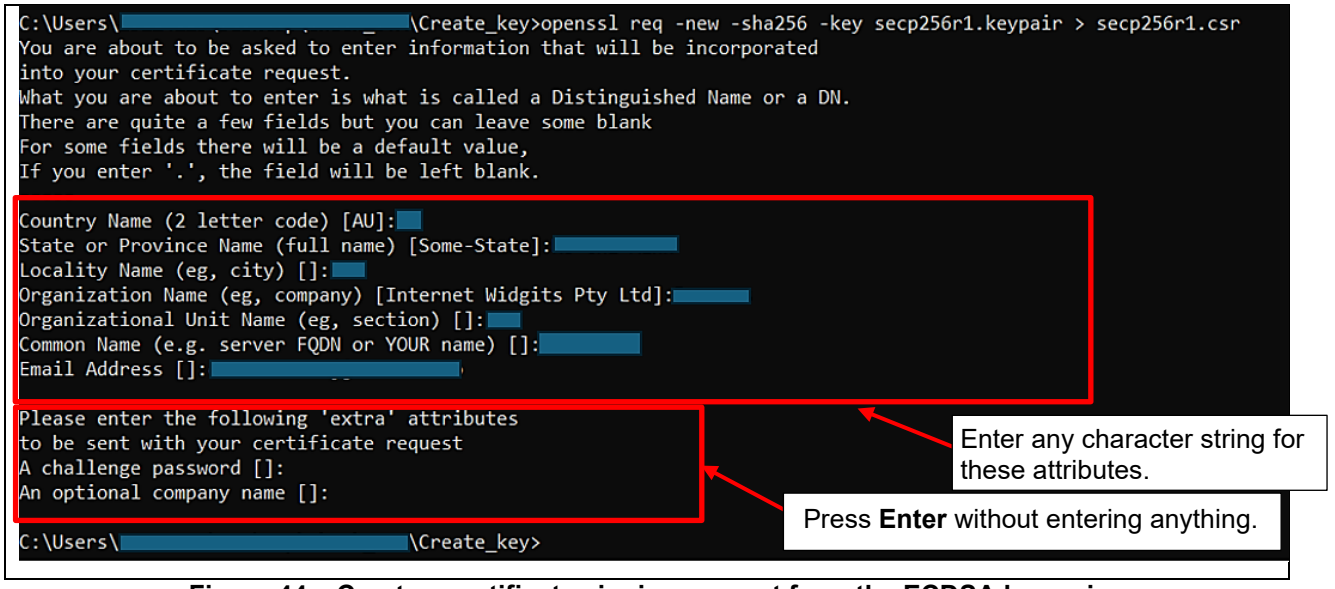

Figure 44. Create a certificate signing request from the ECDSA key pair

(6) Execute the command to create a certificate from the certificate signing request, CA certificate, and CA private key you created

Execute the following command:

"openssl x509 -req -sha256 -days 3650 -in secp256r1.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out secp256r1.crt"

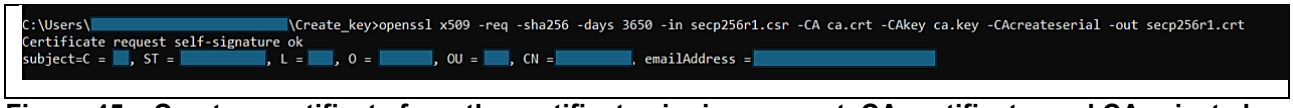

Figure 45. Create a certificate from the certificate signing request, CA certificate, and CA private key

(7) Execute the command to extract the private key from the ECDSA key pair

Execute the following command:

"openssl ec -in secp256r1.keypair -outform PEM -out secp256r1.privatekey"

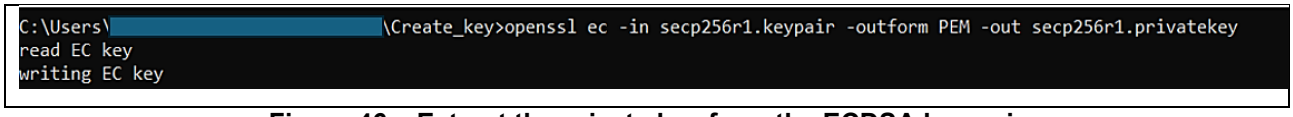

Figure 46. Extract the private key from the ECDSA key pair

(8) Execute the command to extract the public key from the ECDSA key pair

Execute the following command:

"openssl ec -in secp256r1.keypair -outform PEM -pubout -out secp256r1.publickey"

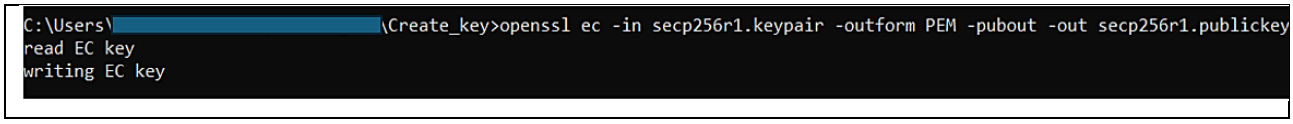

Figure 47. Extract the public key from the ECDSA key pair

3.2.2 Adding the key to the Bootloader project

(1) Replace the file `secp256r1.privatekey` in the folder `bootloader_ck_ra6m5/src` with the generated file from section **3.2.1(7)**.

(2) Open file `bootloader_ck_ra6m5/configuration.xml`. Click the **Generate Project Content** button.

Figure 48. Generate Project Content of the bootloader project

(3) Configuring the Python Signing Environment

Signing the application image can be done using a post-build step in e² studio, using the image signing tool imgtool.py, which is included with MCUboot. This tool is integrated as a post-build tool in e² studio to sign the application image.

If this is the first time you are using the Python script image signing tool on your system, you will need to install the dependencies required for the script to work.

Note: If you have already set up this environment, you can skip this step.

(a) Open Command Prompt from the folder /`MCUboot`

Navigate to the **bootloader_ck_ra6m5>ra>mcutools>MCUboot** folder in the **Project Explorer**, right click, and select **Command Prompt**. This will open a command window with the path set to the `/mcu-tools/MCUboot` folder.

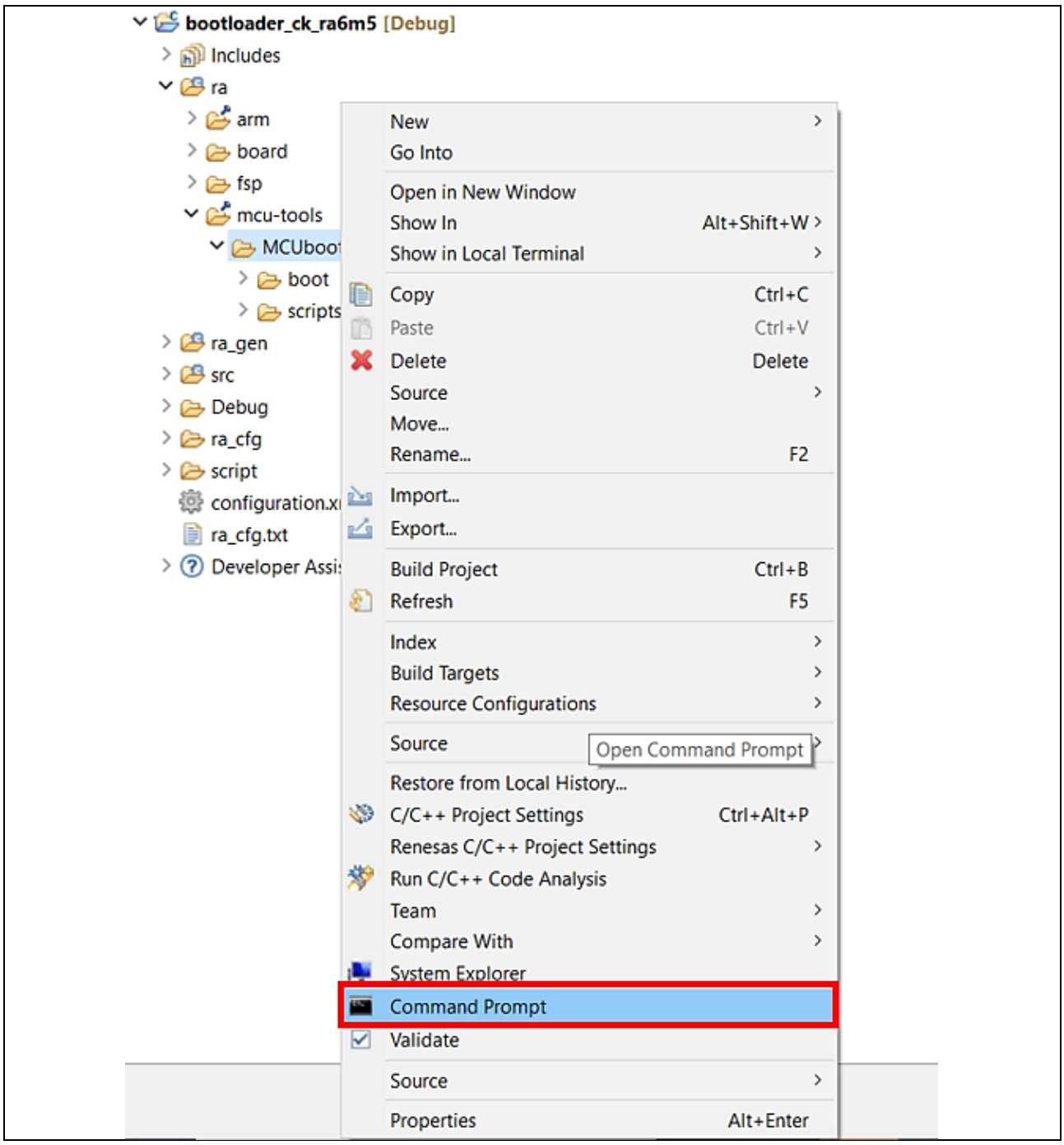

Figure 49. Select Command Prompt from the folder /MCUBoot

(b) Upgrade pip

We recommend upgrading pip prior to installing the dependencies. Enter the following command to update pip:

"`python -m pip install --upgrade pip`"

(c) Install all the MCUboot dependencies

Next, in the command window, enter the following command line to install all the MCUboot dependencies:

"`pip3 install --user -r scripts/requirements.txt`"

This will verify and install any dependencies that are required.

- **Note:** If you encounter an error "No module name pkg_resources" when building the application project. Execute command:
  "`pip install types-pkg-resources`"

(4) Expand `bootloader_ck_ra6m5`, right-click on the folder “ra/mcu-tools/MCUboot/scripts”. Select **Command Prompt** from this folder.

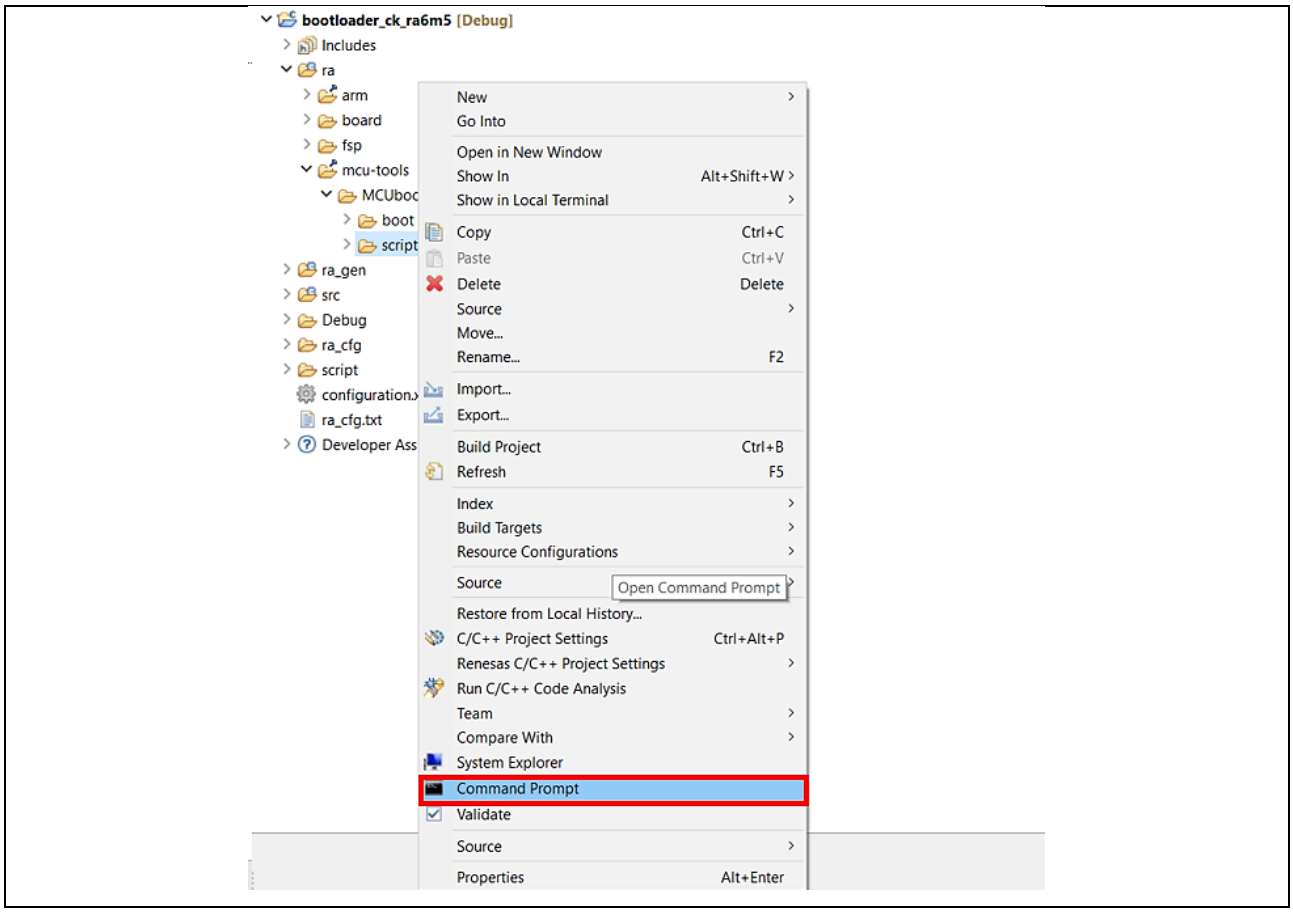

Figure 50. Select Command Prompt from folder /scripts

(5) Execute command

“`python imgtool.py getpub -k ../../../../src/secp256r1.privatekey`”

Figure 51. Get public key from file secp256r1.privatekey

(6) Copy the generated content of `ecdsa_pub_key` to the array `root_pub_der` in file `src/keys.c`. Replace the original `root_pub_der` content. (just replace the value in the array, the length of the array)

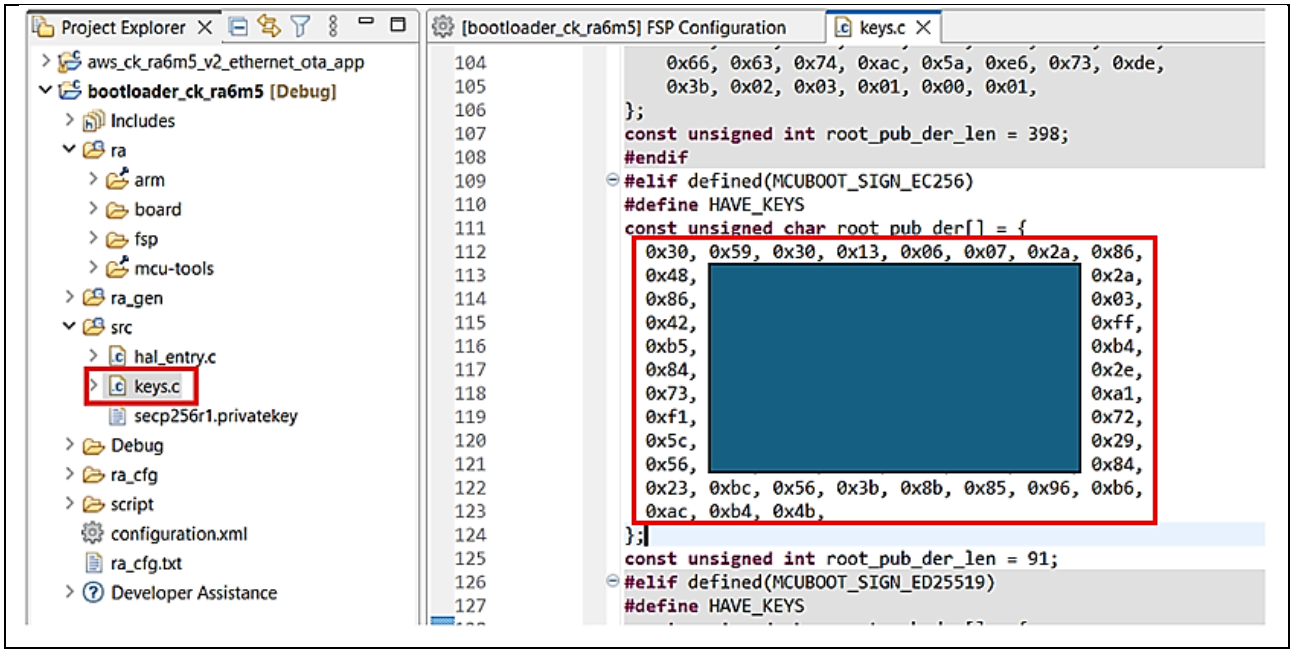

Figure 52. Replace root_pub_der content

3.3 Building all the projects

Right click to `aws_ck_ra6m5_v2_ethernet_ota_solution` **> Build Project:**

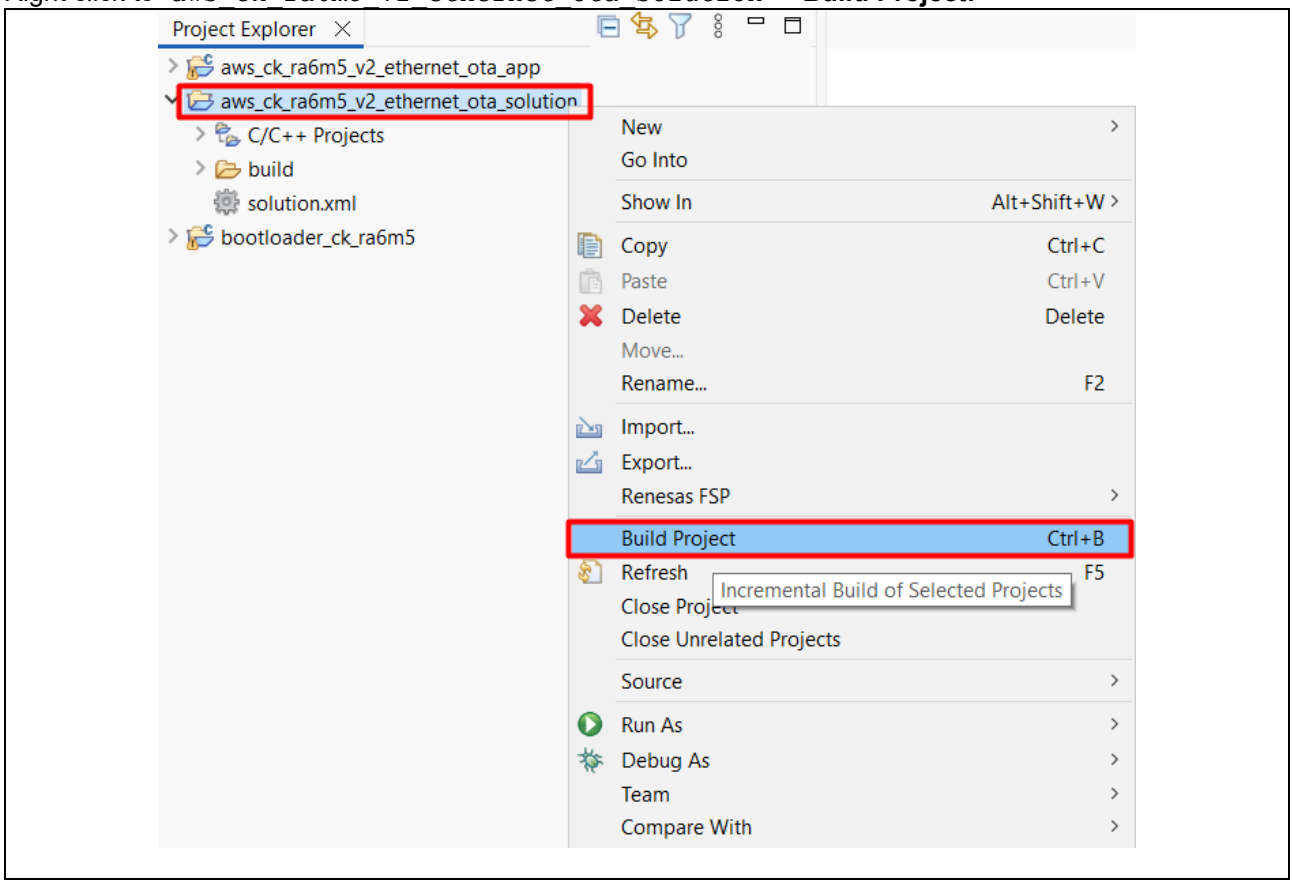

Figure 53. Build aws_ck_ra6m5_v2_ethernet_ota_solution project

Since the FSP solution project is linked with the bootloader and application project, when building the FSP solution project, the bootloader project and application project will be built sequentially. The bootloader project is built first, followed by the application project. This is based on the memory layout in the configuration of the FSP solution project.

After the projects were built successfully, the signed image is located under the `\Debug` folder: `aws_ck_ra6m5_v2_ethernet_ota_app\Debug\aws_ck_ra6m5_v2_ethernet_ota_app.bin.signed`

3.4 Connection Settings and Deviation

Reset the board assembly associated with this application note to the default electrical jumper settings as specified in the *CK-RA6M5 v2 User's Manual* before proceeding with the next set of instructions.

Note: For this Ethernet-based cloud connectivity application project and application note, the user is required to connect the Ethernet cable to the connector (J5) on the board.

3.5 Powering up the Board

To connect power to the board, connect the USB cable to the CK-RA6M5 v2 board's J28 connector (USBC) and the other end to the PC USB port. Connect the second USB Cable to the J10 (USB_DBG) connector of the CK-RA6M5 v2 board and the other end to the second USB Port of the PC (This will be the Console Port for Application). Users are required to use the Command Line Interface (CLI) with J-Link Virtual COM port to configure and run the Application.

3.6 Setting Trustzone Boundary for the Board

The design guidelines in this section are specific to some of the Arm® TrustZone® MCUs, which need special configurations to access the non-secure buffers.

For the RA MCU with EDMAC and Arm TrustZone support, the EDMAC peripheral can only be configured with the non-secure attribute, which is described by the Peripheral Security Attribution section in the User's Manual. Therefore, the EDMAC on these target devices always requires the non-secure buffers to be located in a non-secure attributed RAM area.

In the e^2 studio, choose to debug from the application project `aws_ck_ra6m5_v2_ethernet_ota_app`.

Right-click on project `aws_ck_ra6m5_v2_ethernet_ota_app` and select **Debug As > Debug Configurations> Renesas GDB Hardware Debugging** drop-down list. Select **aws_ck_ra6m5_v2_ethernet_ota_app Debug_Flat > Debugger > Connection Setting > TrustZone > Set TrustZone secure/non-secure boundaries > No** and **"Apply"** the setting.

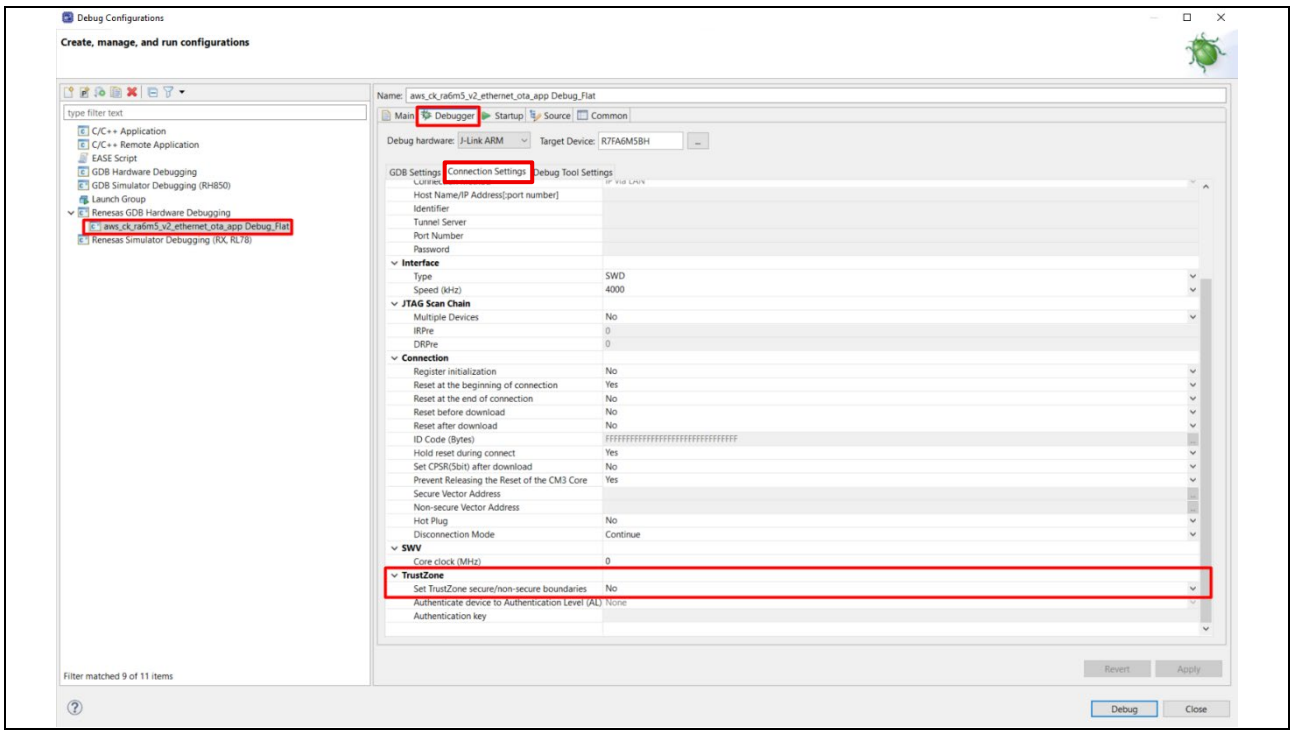

Figure 54. Set TrustZone secure/non-secure boundaries to “No” in debug configuration

After that, set TrustZone boundaries manually by using Renesas Device Partition Manager.

In e^2 studio, go to **Run > Renesas Debug Tools > Renesas Device Partition Manager**

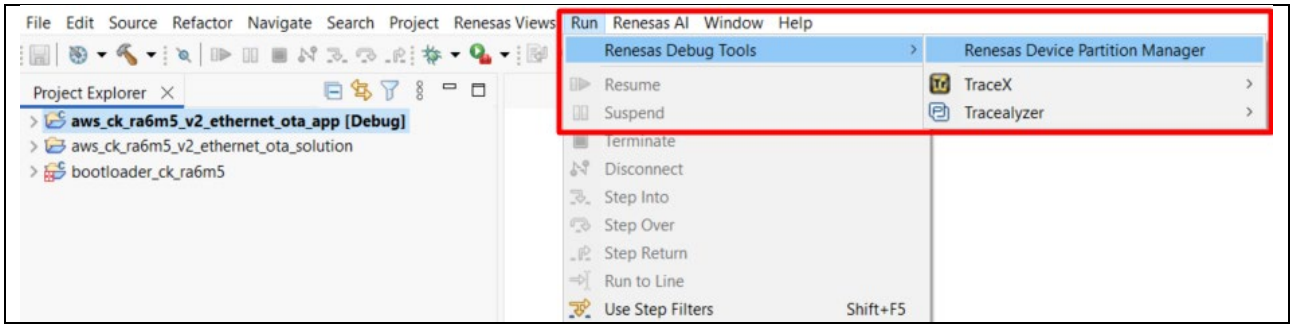

Figure 55. Open Renesas Device Partition Manager

Choose “**Initialize device**” then click **Run** as shown in the figure below.

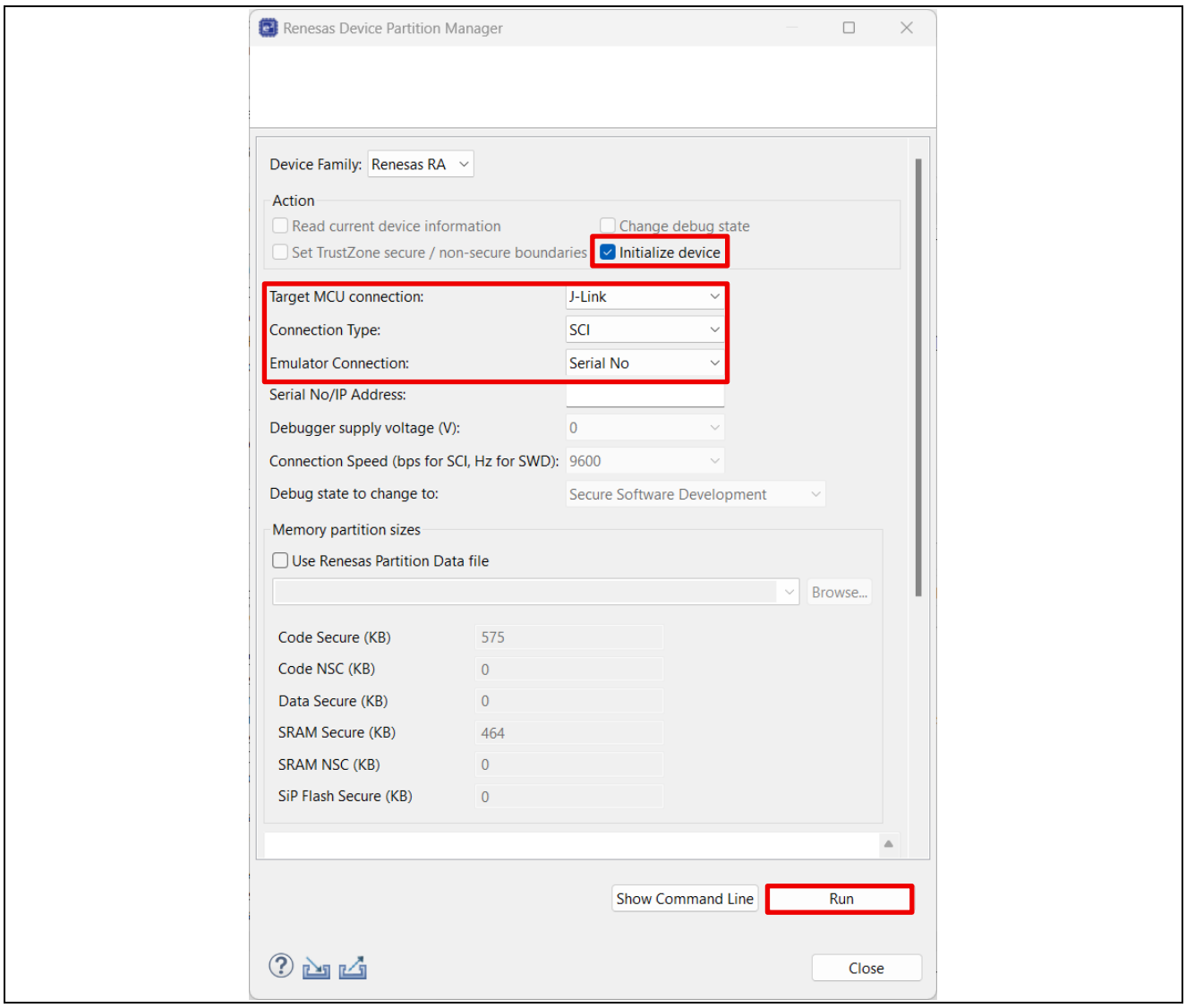

Figure 56. Initialize device

Choose **"Set TrustZone secure / non-secure boundaries"** and set Renesas Device Partition Manager as shown in the figure below. Click **Run** to set TrustZone boundaries.

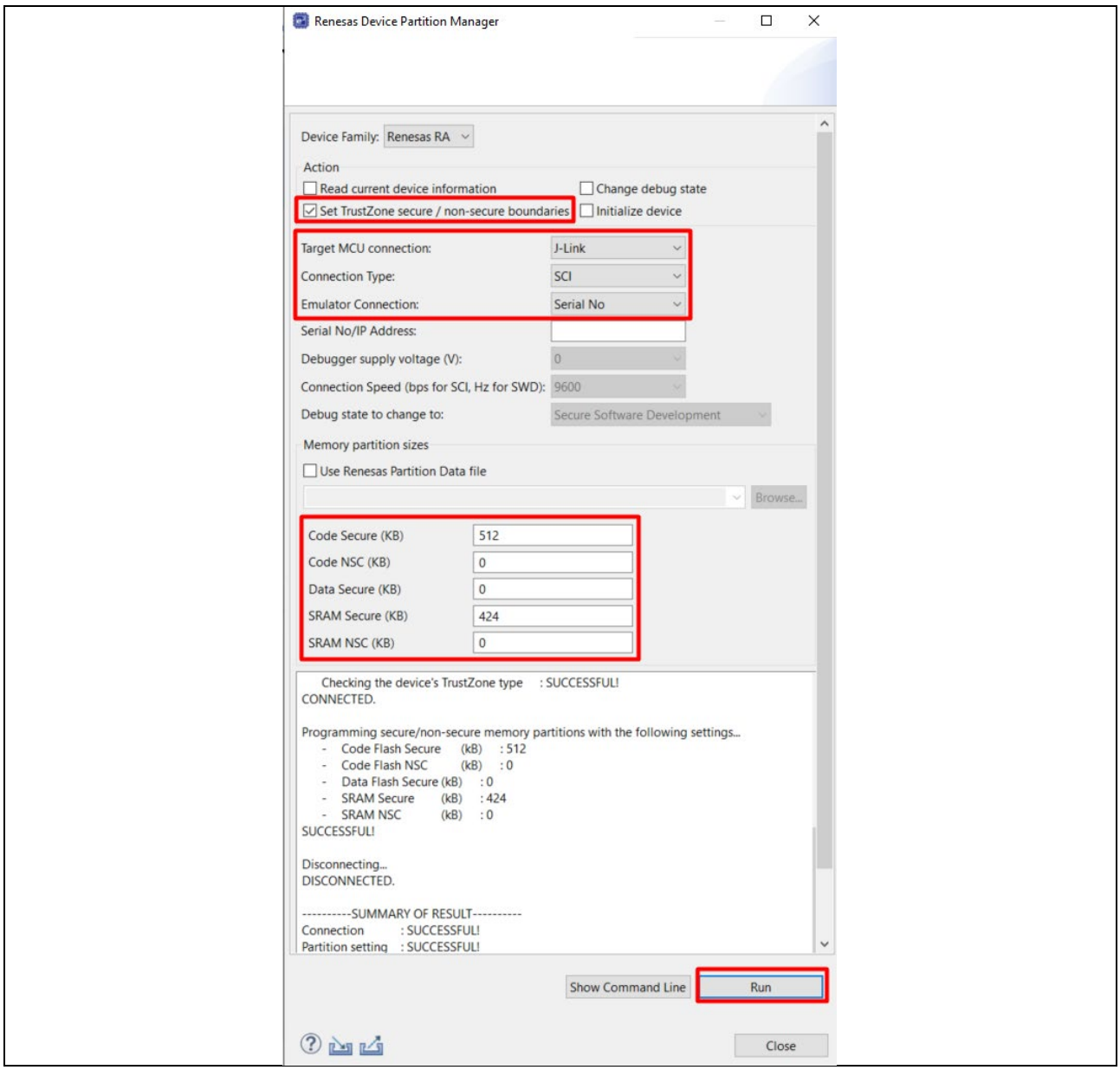

Figure 57. Setting TrustZone boundaries with Renesas Device Partition Manager

3.7 Loading the Applications

Choose to debug from the application project `aws_ck_ra6m5_v2_ethernet_ota_app`.

Right-click on project `aws_ck_ra6m5_v2_ethernet_ota_app` and select **Debug As > Debug Configurations > Renesas GDB Hardware Debugging drop-down list**. Select **aws_ck_ra6m5_v2_ethernet_ota_app.elf > Startup** and confirm that the following configuration exists.

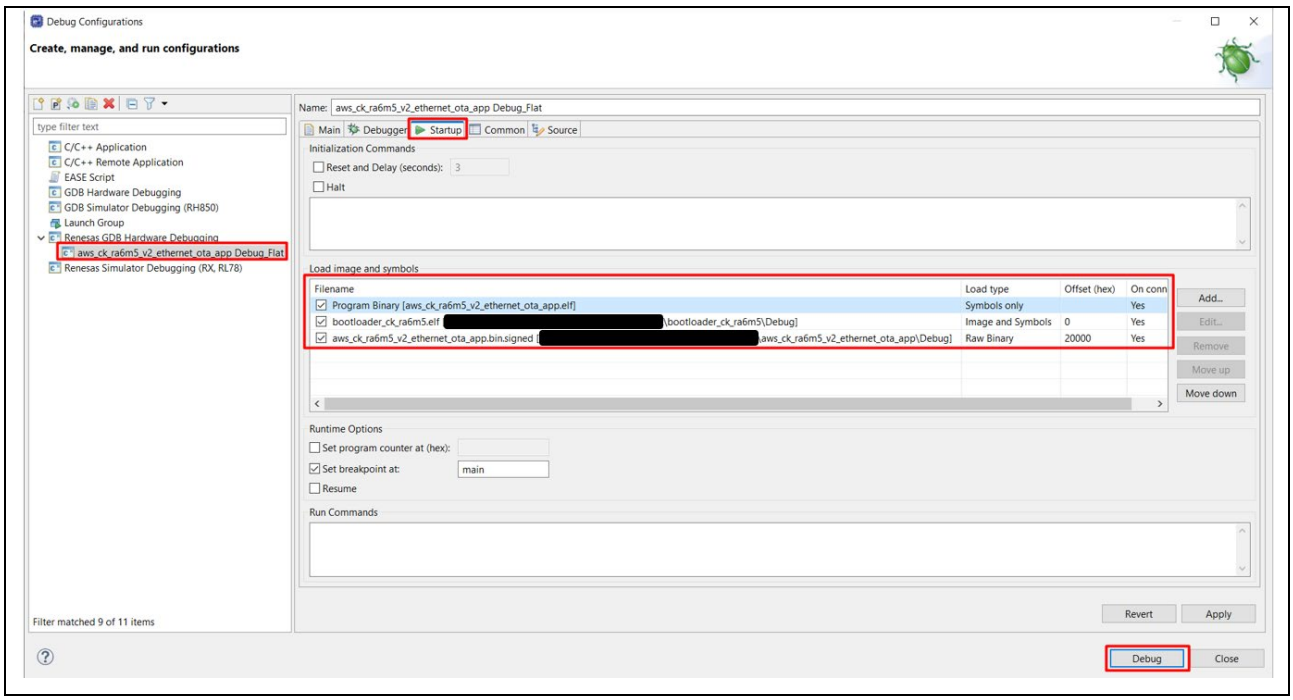

Figure 58. Debug Configurations

- Under the Startup configuration, verify the Load type of `aws_ck_ra6m5_v2_ethernet_ota_app.elf` is Symbols only.
- The `aws_ck_ra6m5_v2_ethernet_ota_app.bin.signed` entry exists with the Load type as Raw Binary, and the Offset is set to 0x20000 since that is the initial firmware of the application project `aws_ck_ra6m5_v2_ethernet_ota_app`.
- The `bootloader_ck_ra6m5.elf` is added with Load type as Image and Symbols with an Offset of 0 since the bootloader starts from 0x0.

Note: If the above configuration has not been set up, please follow each step below:

- Click **Add**… and then **Workspace**. Navigate to the **`bootloader_ck_ra6m5`** project and select the **`bootloader_ck_ra6m5.elf`** file from the debug folder. Click **OK**.

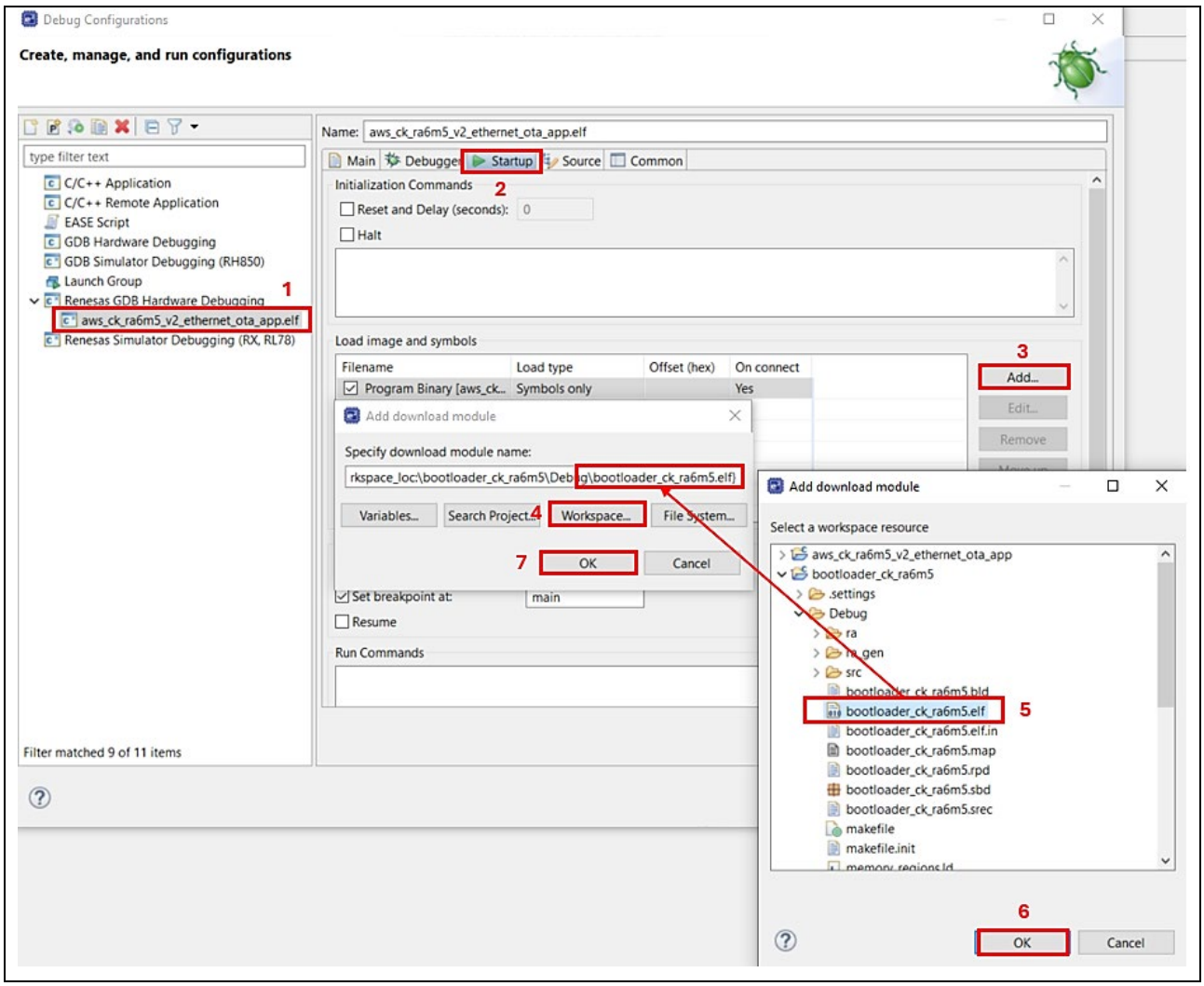

Figure 59. Add the Bootloader project to the Debug Configuration

- Click **Add** again and add the **`aws_ck_ra6m5_v2_ethernet_ota_app`** project binary **`aws_ck_ra6m5_v2_ethernet_ota_app.bin.signed`** as in the prior step. Click **OK**.

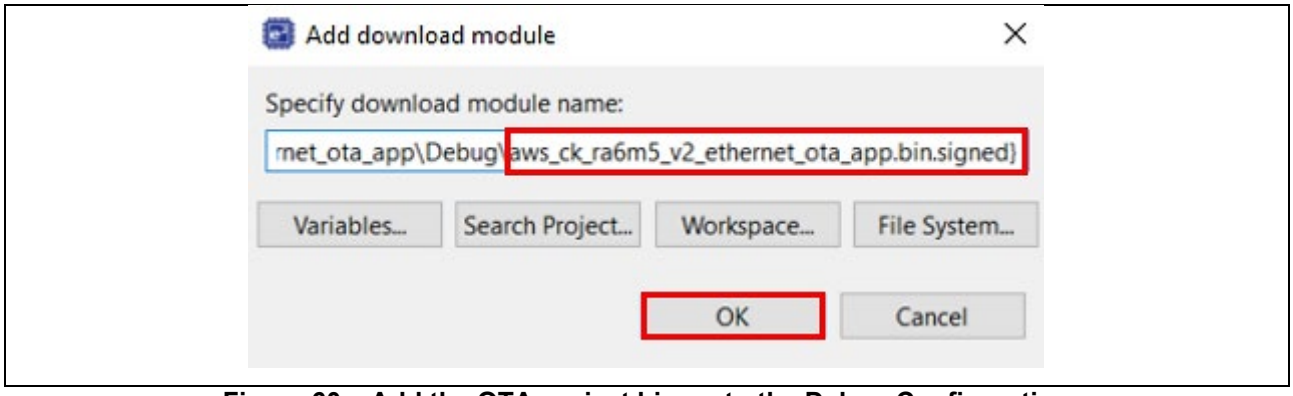

Figure 60. Add the OTA project binary to the Debug Configuration

- Change the **Load type** of the Program Binary for the **`aws_ck_ra6m5_v2_ethernet_ota_app`** by clicking on the cell for **Load type** and selecting **Symbols only** from the drop-down menu. Similarly, change the load type of the signed binary image **`aws_ck_ra6m5_v2_ethernet_ota_app.bin.signed`** to **Raw Binary** Load type, and the **Offset (hex)** is set to **20000**; **`bootloader_ck_ra6m5.elf`** with **Image and symbols**, and the **Offset (hex)** is set to **0**. Please refer to Figure 58. Debug Configurations to verify the setting result.

Click **Debug**, then **Resume** the execution twice by clicking  .

4. Running the Application Project

Note: The steps indicated below are tested on Windows OS only.

4.1 Connecting the Board to the Serial Port 7v Console of the PC

1. On the host PC, open Windows Device Manager. Expand **Ports (COM & LPT)**, locate **JLink CDC UART Port (COMxx)**, and note down the COM port number for reference in the next step.

   Note: JLink CDC UART drivers are required to communicate between the CK-RA6M5 v2 board and the terminal application on the host PC.

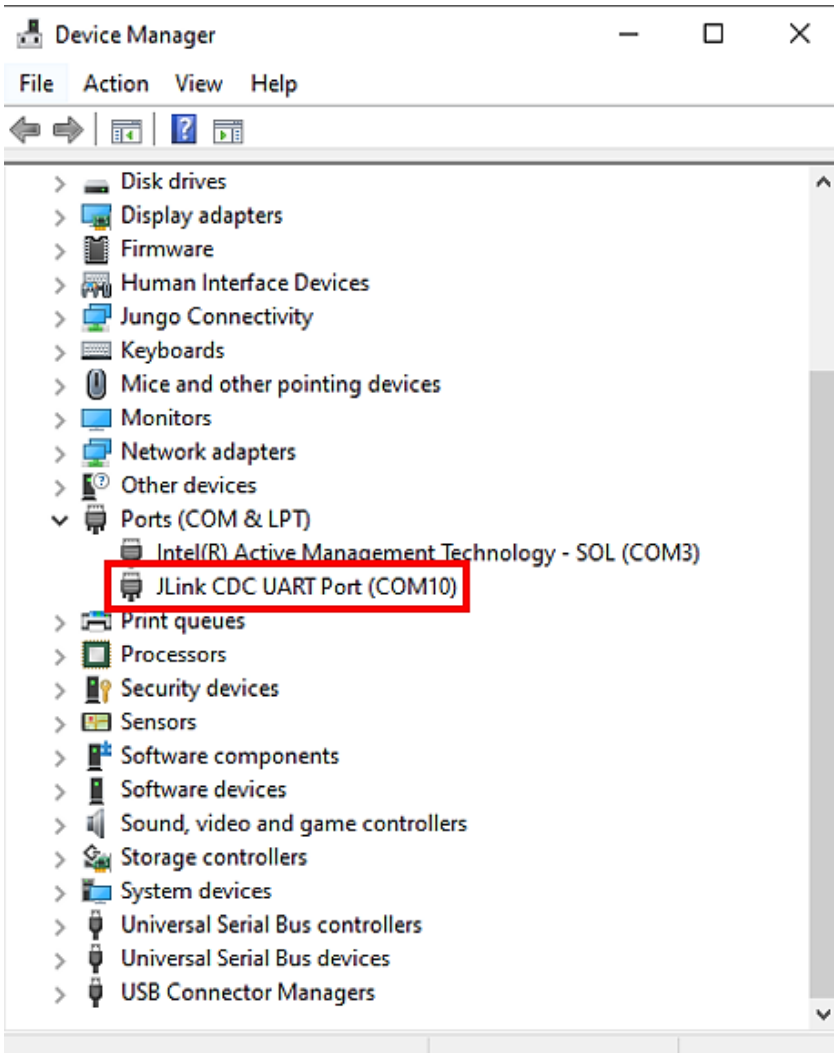

Figure 61. JLink CDC UART in Windows Device Manager

2. Open Tera Term, select **New connection**, select **Serial** and **COMxx: JLink CDC UART Port (COMxx),** and click **OK**.

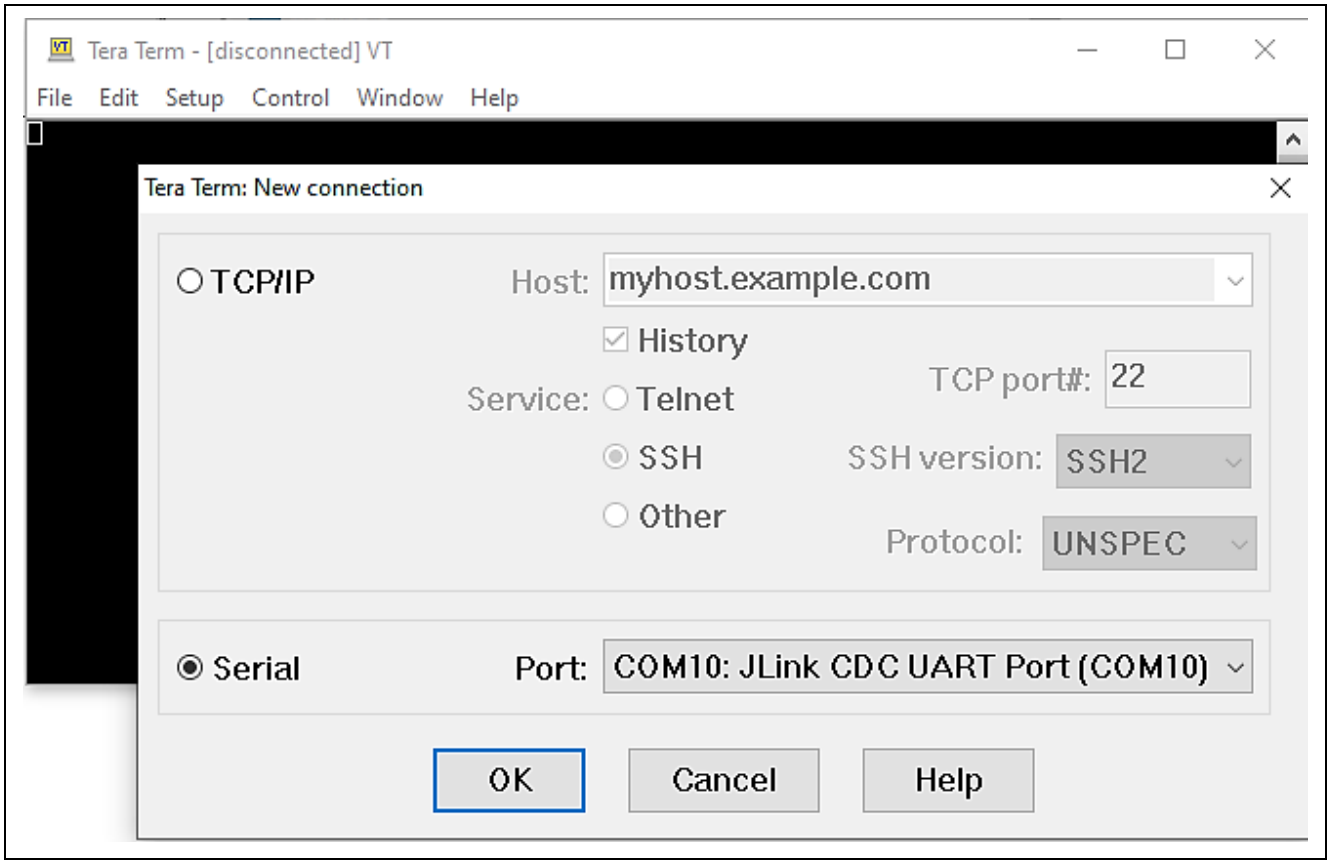

Figure 62. Selecting the UART Port on Tera Term

3. Make sure Tera Term selects the black background; if not, configure it from **Setup > Window** and make the following selections.

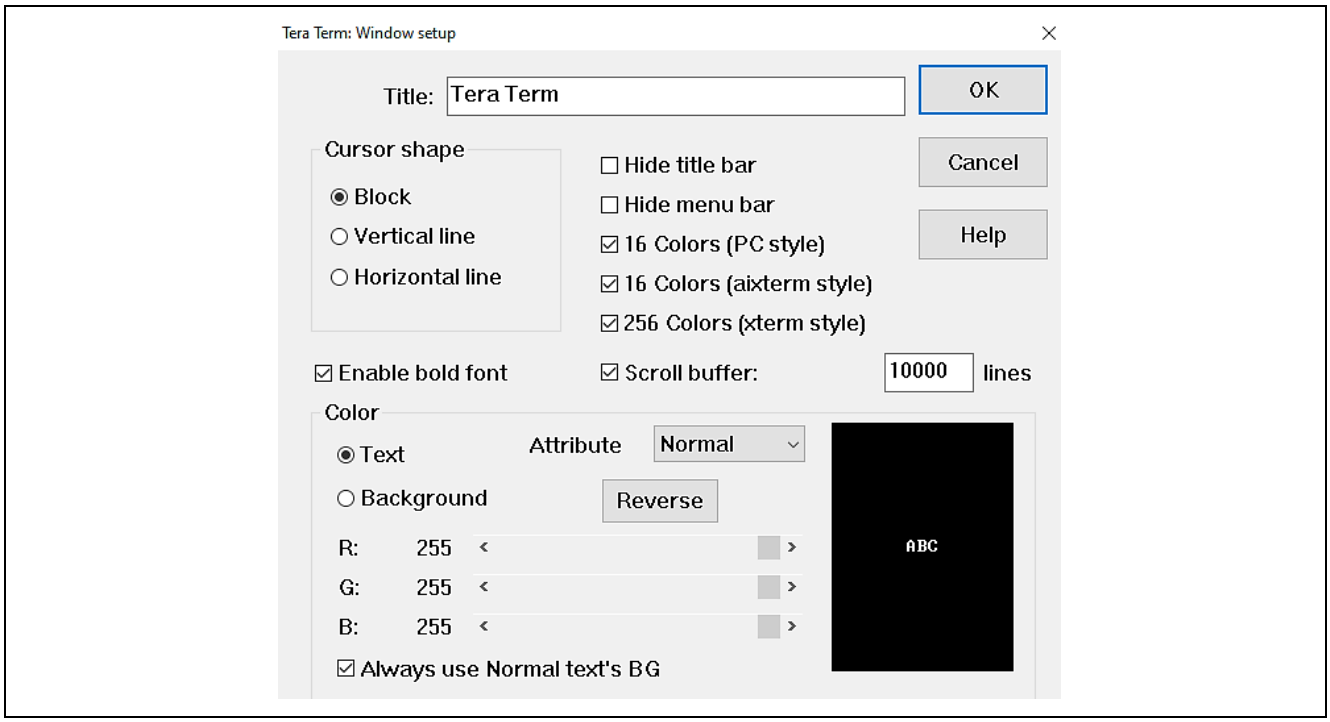

Figure 63. Configuring the Black Background for JLink CDC UART Port on Tera Term

4. Configure for the terminal from **Setup > Terminal…**, select **New-line Receive** as **AUTO**.

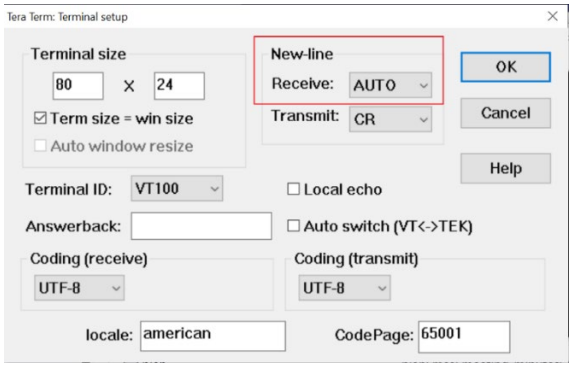

Figure 64. Configuring Terminal

5. Using the **Setup > Serial port…** and ensure that the speed is set to **115200**, as shown below.

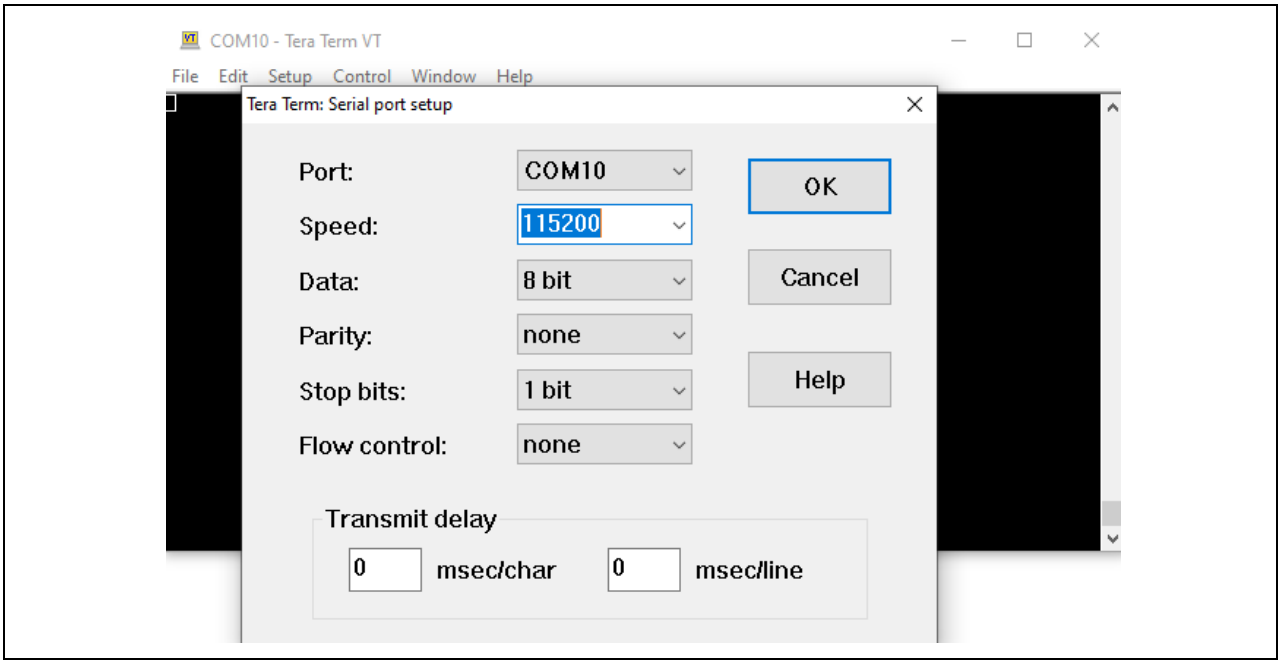

Figure 65. Select 115200 on the Speed Pulldown Menu

6. Complete the connection. Please reset by pressing the **S1** user switch of the CK-RA6M5 v2 board. When the red message below appears, press any key to go to the application's settings area. If not, the application will run automatically.

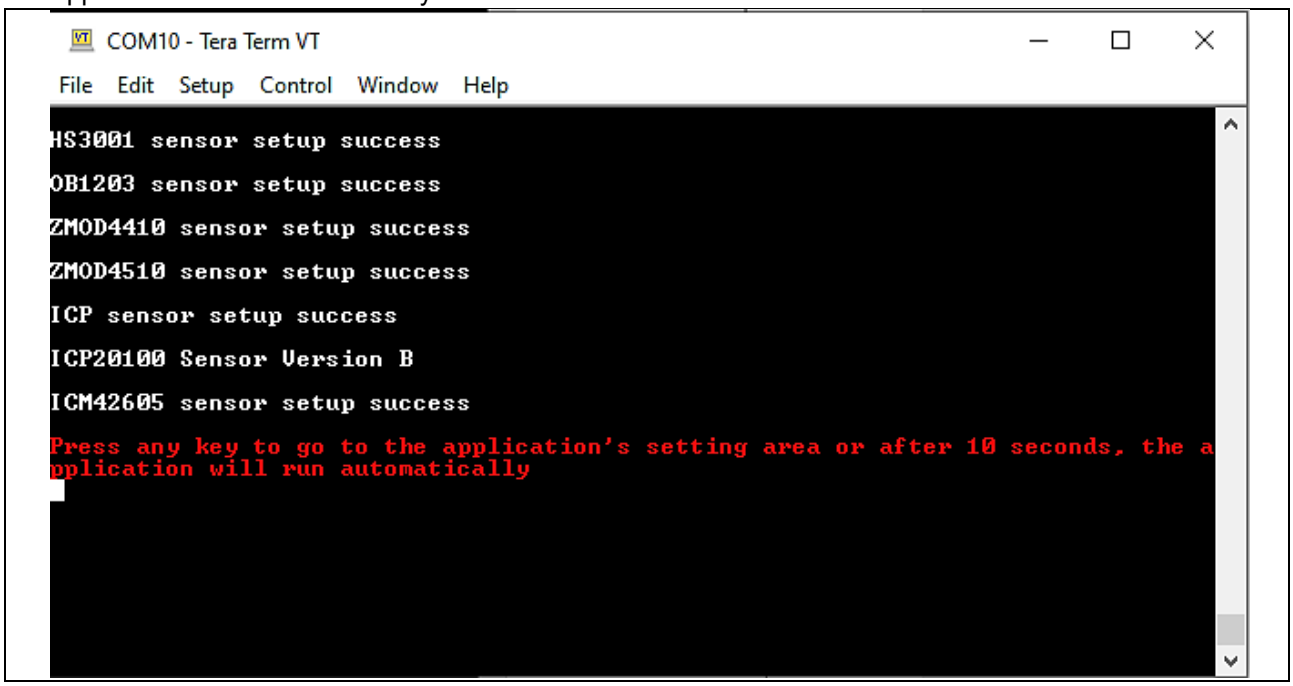

Figure 66. Press any key to go to the application's settings area

The Configuration CLI Menu will be displayed on the console as shown below.

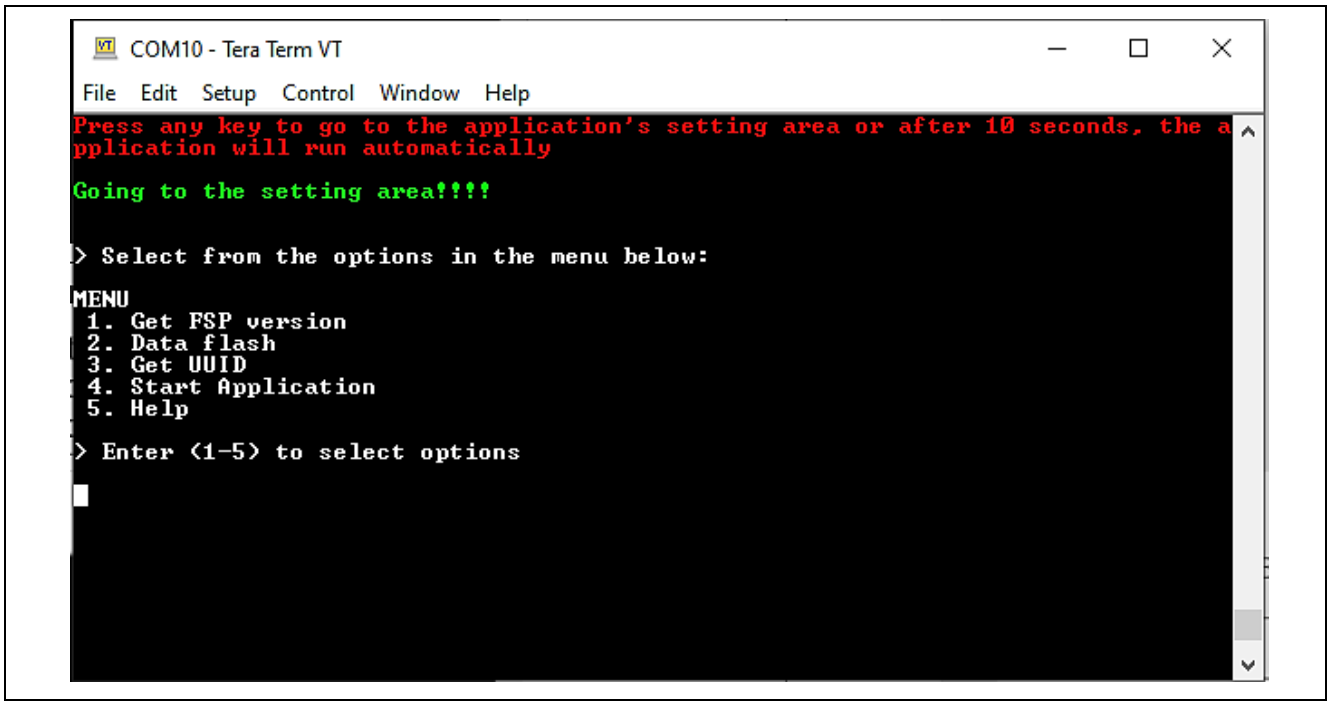

Figure 67. Main Menu

7. In the CLI shown in the preceding screenshot, choose the number to select the commands. For example, when you press **1**, the FSP version of the application is displayed below. At any point in time, press the space bar to return to the previous menu.

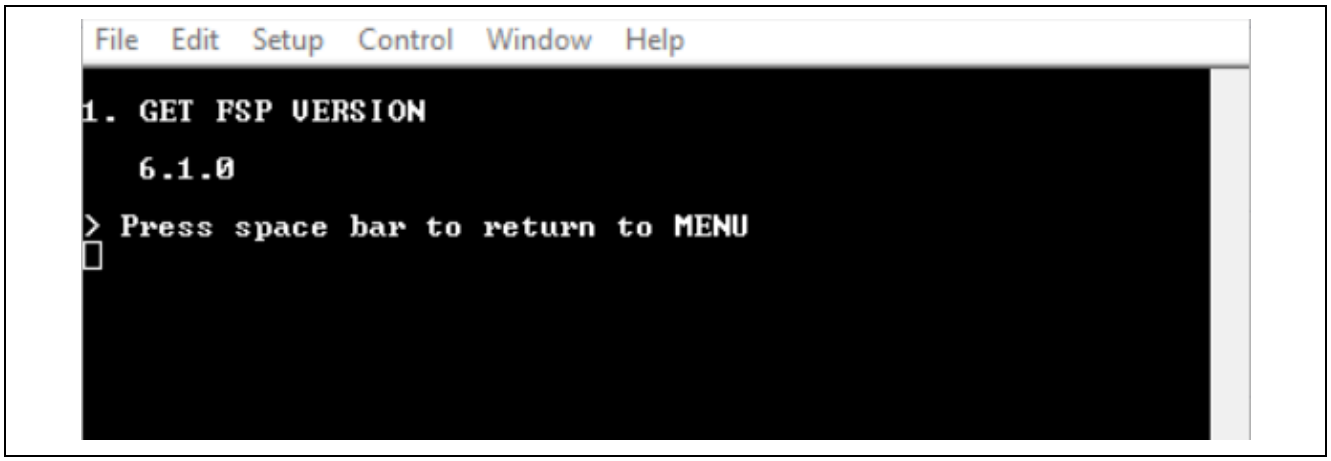

Figure 68. FSP Version Information

4.2 Storing the Device Certificate, Key, MQTT Broker Endpoint, and IoT Thing Name, Code Signing Public Key

Device Certificate, Device Private Key, MQTT Broker Endpoint, IOT Thing name, and Code Signing Public Key need to be stored in the data flash for the application to work.

1. Press **2** on the **Main Menu** to display **Data Flash-related** commands, as shown in the following screenshots. This sub-menu has commands to store, read, and validate the data.

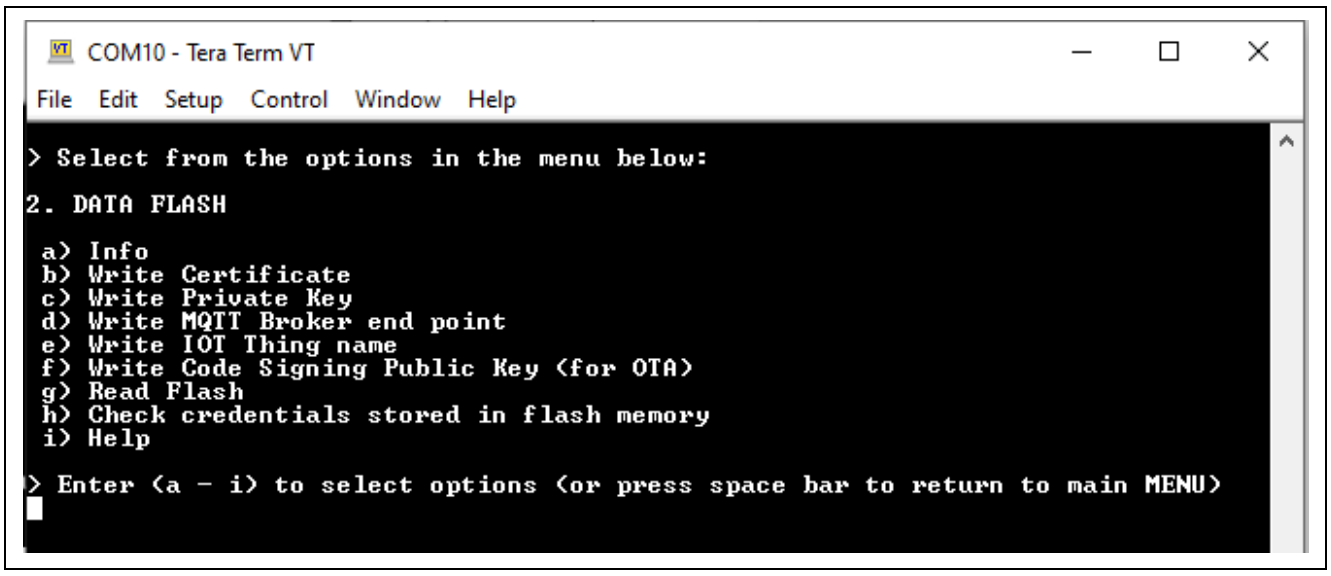

Figure 69. Data Flash-related Menu and Commands

2. To store the **Device Certificate**, press option **b,** click the **File** tab of the Tera Term and **Send File** option, and choose the device certificate file '`xxxxx-certificate.pem.crt`' from the downloaded file in section **2.1.3.2(6)**.

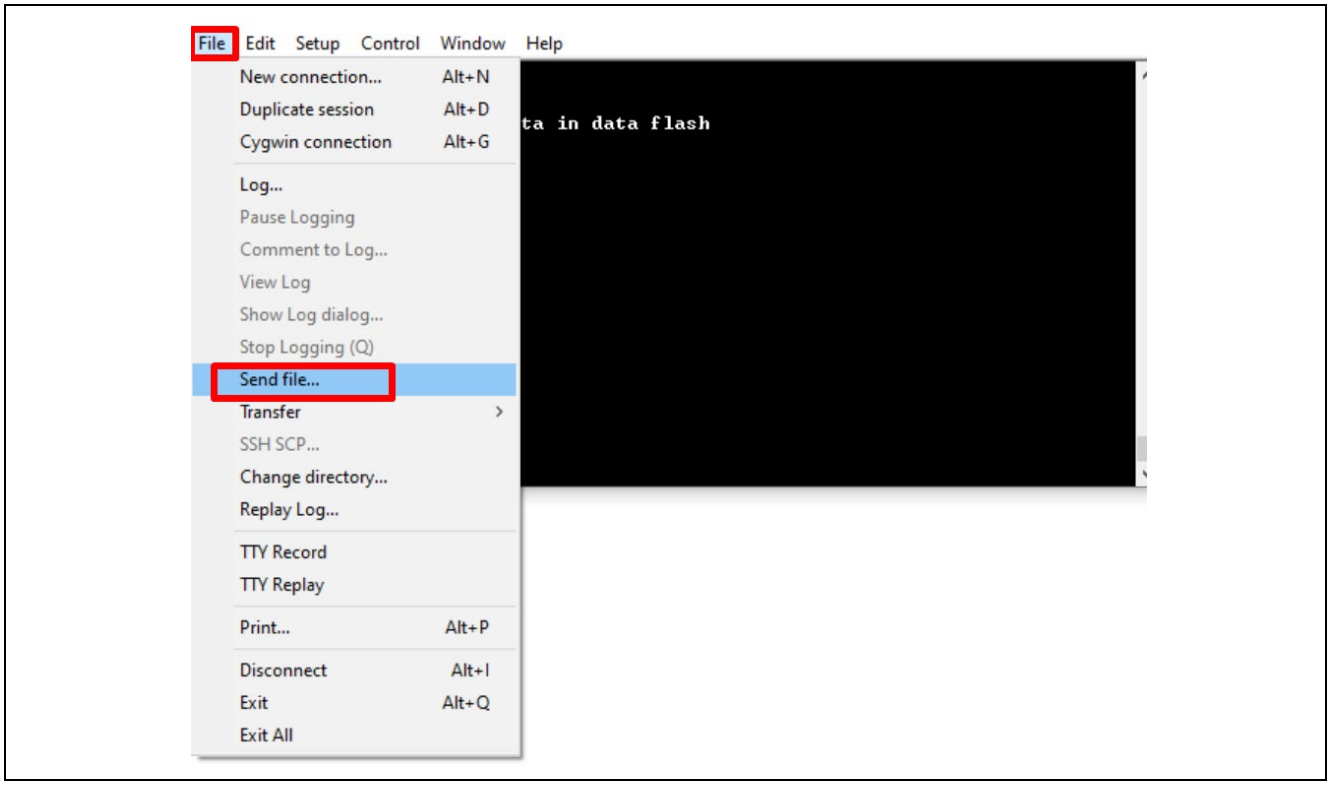

Figure 70. Accessing the Device Certificate

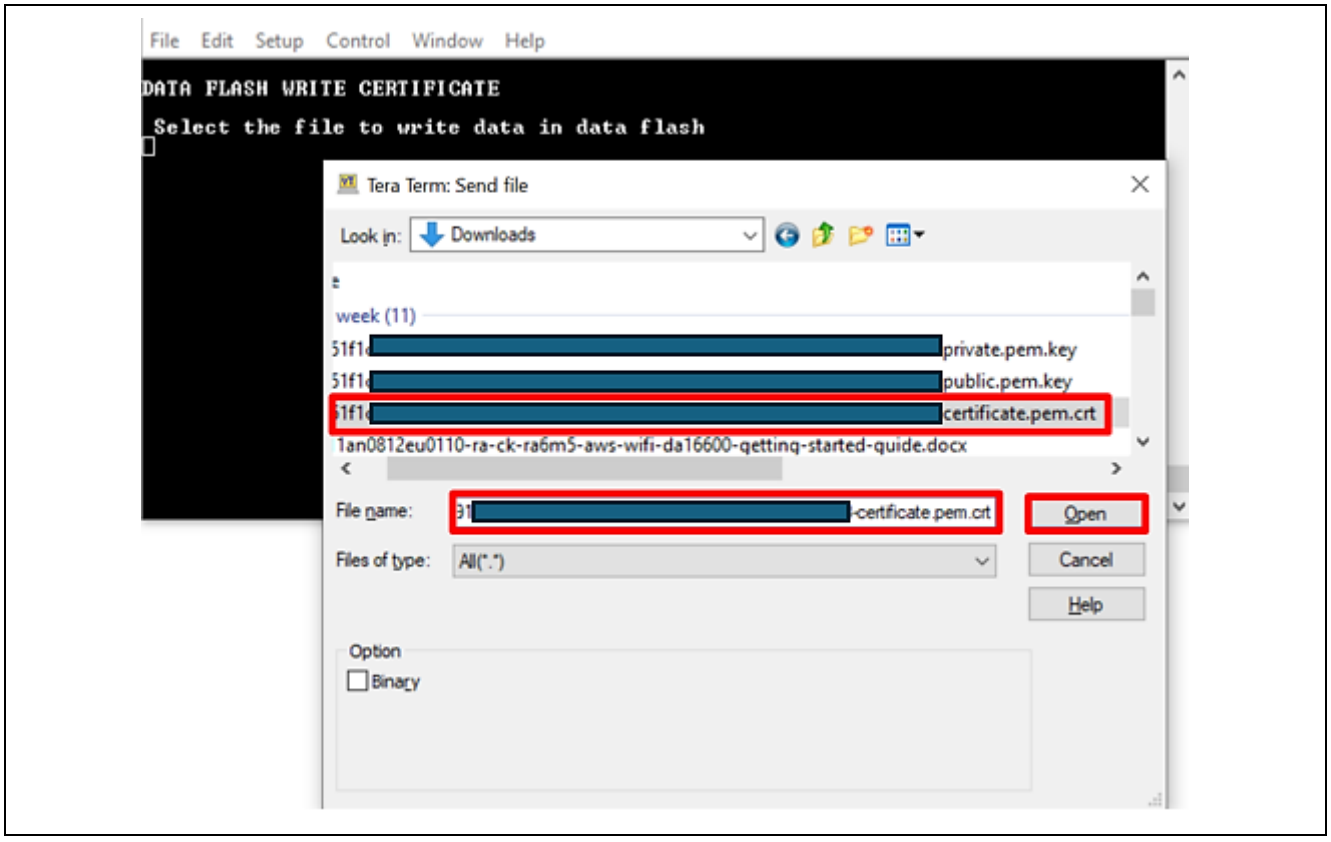

Figure 71. Downloading the Device Certificate into the Data Flash

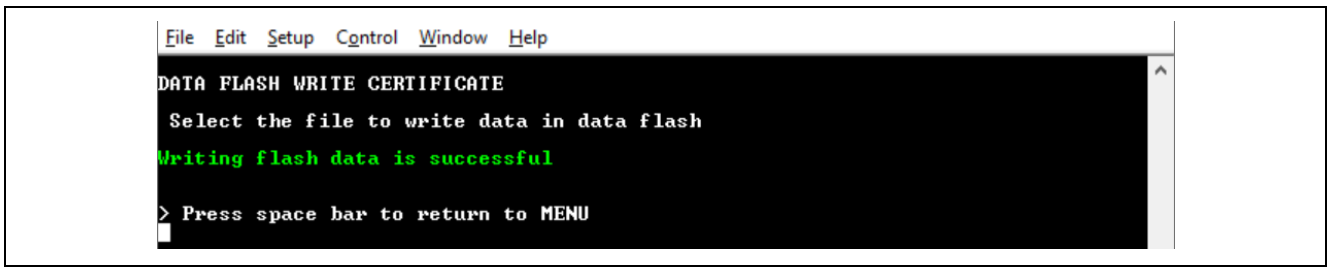

Figure 72. Status of the Downloaded Device Certificate in the Data Flash

3. To store the **Device Private Key,** press option **c** and click the **File** tab of the Tera Term. Select the **Send File** option and choose the Device Private Key "`xxxxxxx-private.pem.key`" from the downloaded file in section **2.1.3.2(6)**.
4. To store the MQTT Broker endpoint**,** copy the endpoint string `xxxxxxxxxxxx-ats.iot.us-east-1.amazonaws.com` in section **2.1.3.3**. Press option **d** and click the **Edit** tab of the Tera Term and **Paste<CR>**. Verify and confirm the valid string and press **OK.**

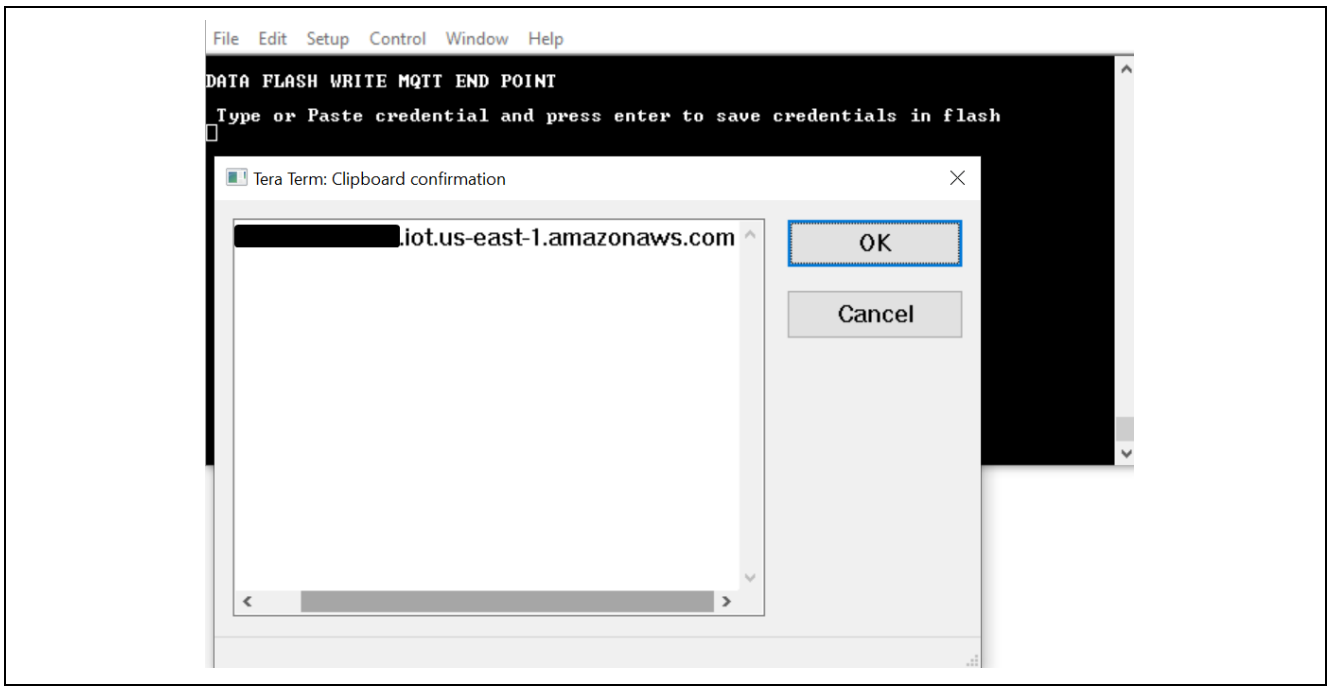

Figure 73. Storing the MQTT Endpoint in the Data Flash

5. To store the IOT Thing Name, copy the Thing Name that was created in section **2.1.3.2(3)**. Press option **e** and click the **Edit** tab of the Tera Term and **Paste<CR>**. Verify and confirm the string is valid, then press **OK.**

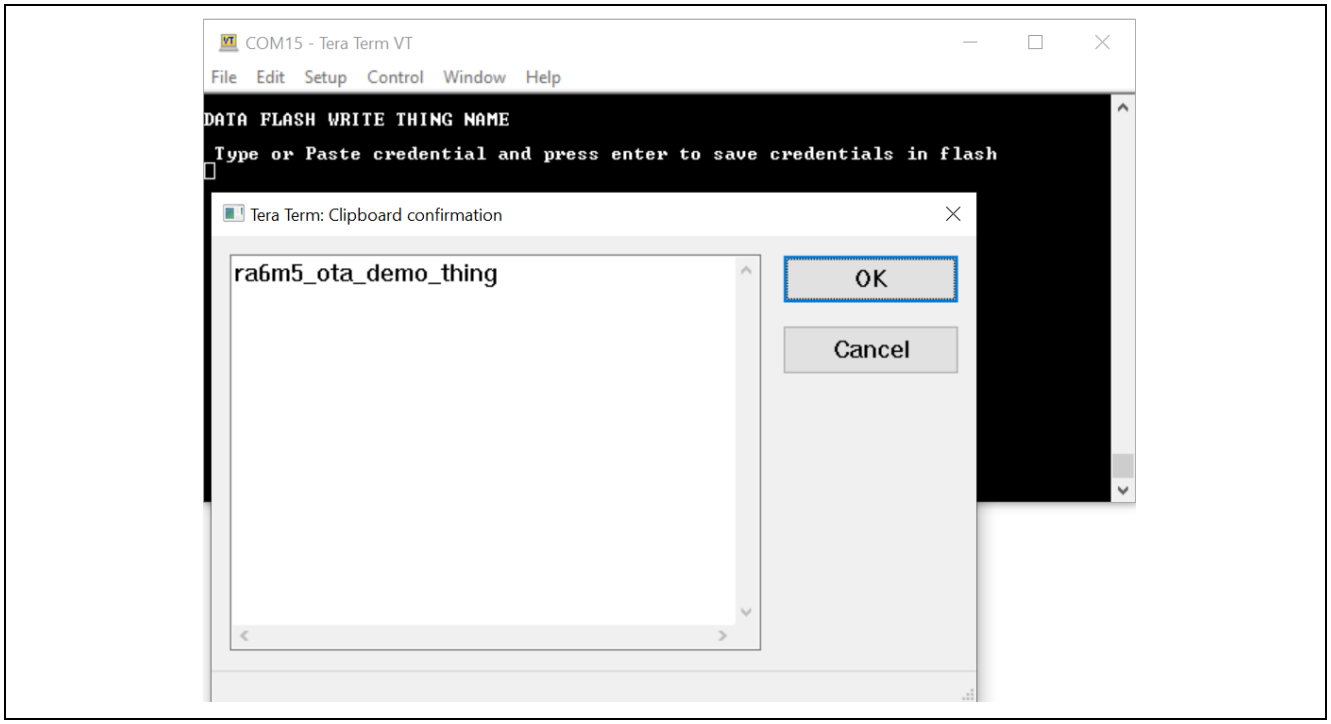

Figure 74. Storing the MQTT Endpoint in the Data Flash

6. Store the **Code Signing Public Key**

Note: Please change the end of the line of the file `secp256r1.publickey`, which is created in section **3.2.1(8)**, to Unit (LF) type. If there is no change, the application can't switch to the new firmware when receiving a new firmware image.

Below is an example of converting secp256r1.publickey, which is created in section **3.2.1(8),** to Unit (LF) type using Notepad++.

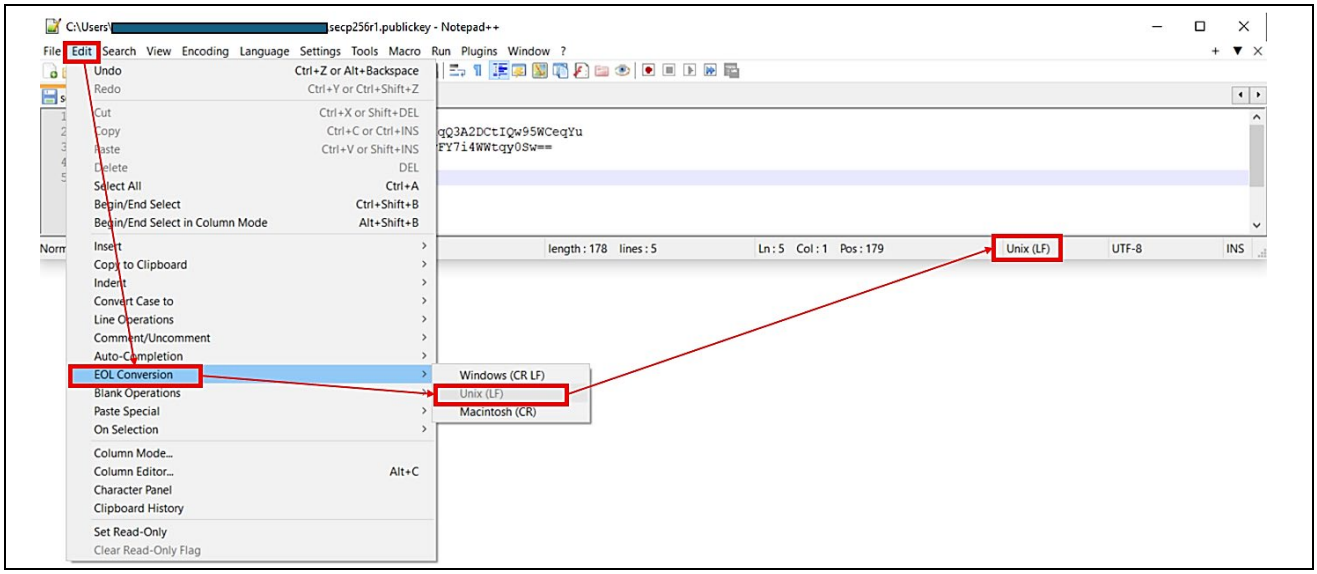

Figure 75. Change the end of the line of file secp256r1.publickey to Unit (LF) type using Notepad++

To store the **Code Signing Public Key**, press option **f**, click the File tab of the Tera Term, and the Send File option, and choose the file `secp256r1.publickey`, **which is changed to Unit (LF) type**.

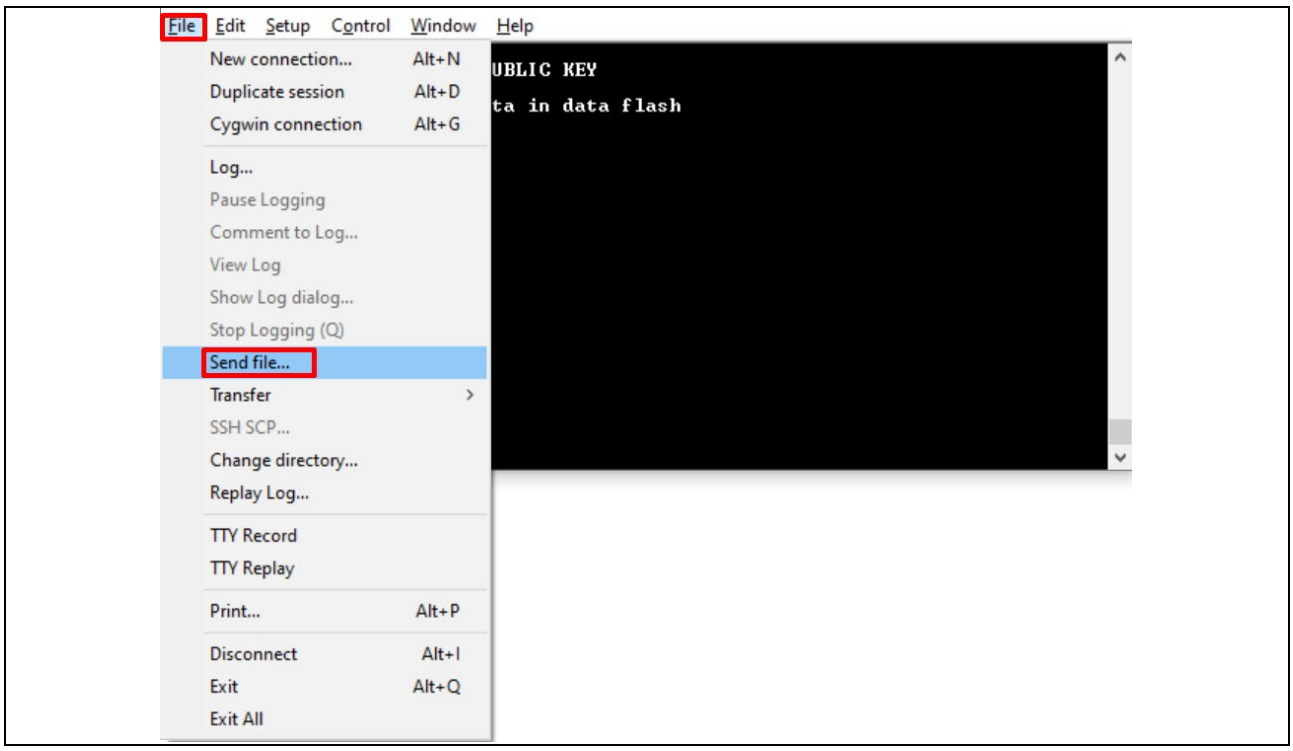

Figure 76. Accessing the Code Signing Public Key

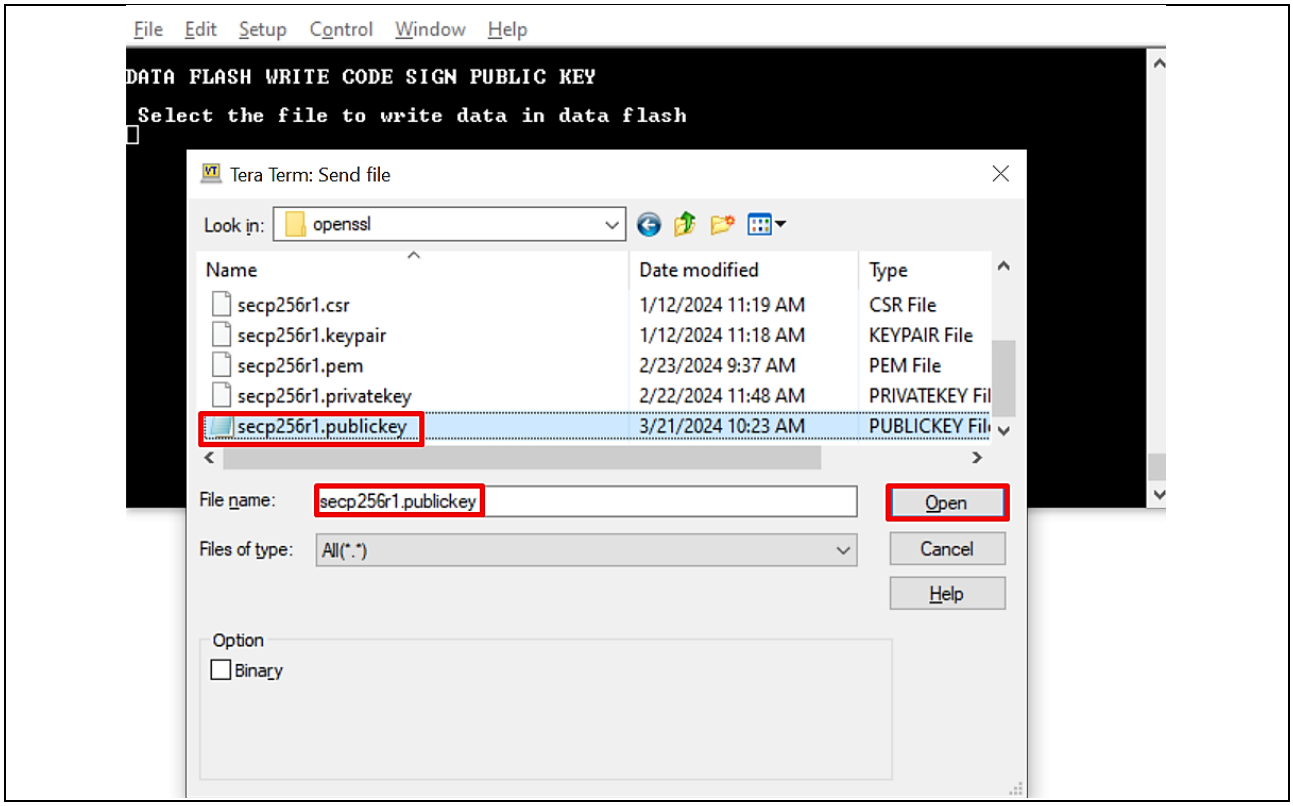

Figure 77. Downloading the Code Signing Public Key into the Data Flash

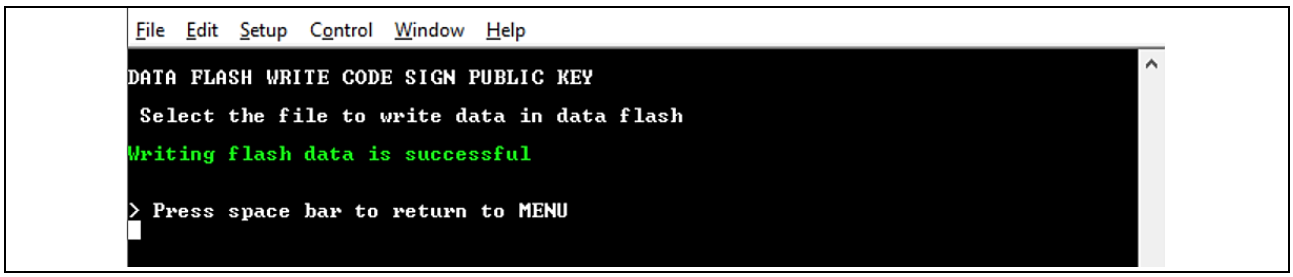

Figure 78. Status of the Code Signing Public Key into the Data Flash

7. Press options **g** and **h** to read and validate the stored information in the data flash. Press the space bar to go to the previous menu.

   Note: Validation of the stored data is very limited and validates a minimum set of data points. Users are required to input the valid data into the flash obtained from the AWS (Cloud credentials) for the proper working of the application.

4.3 Starting the Application

After configuring the required Cloud credentials through the CLI, the application is ready to run. Press option "**4. Start Application"** to start the application. The application prints a Welcome screen along with the status of validating the Cloud credentials data present in the data flash, as shown below.

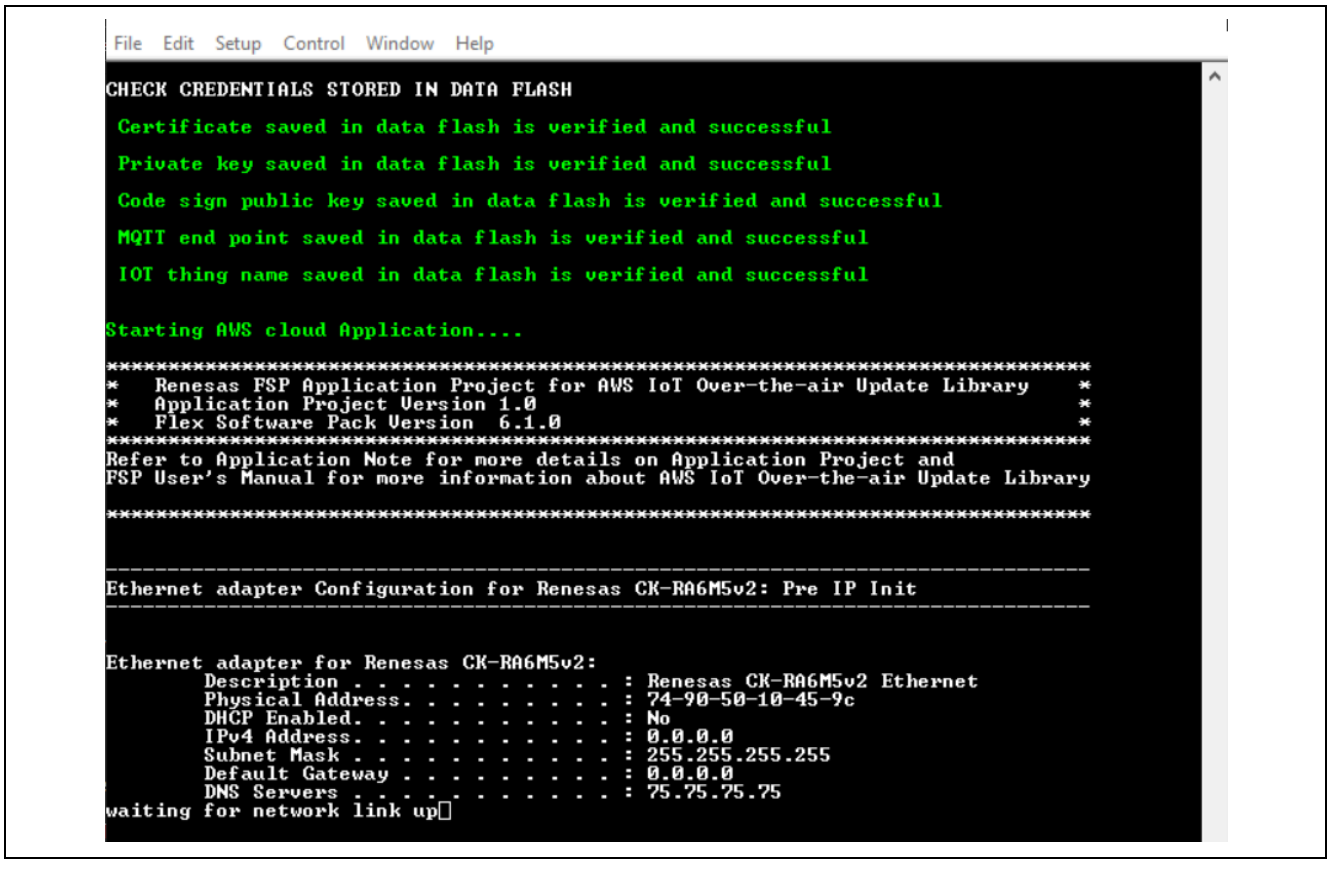

Figure 79. Welcome Screen on the Console

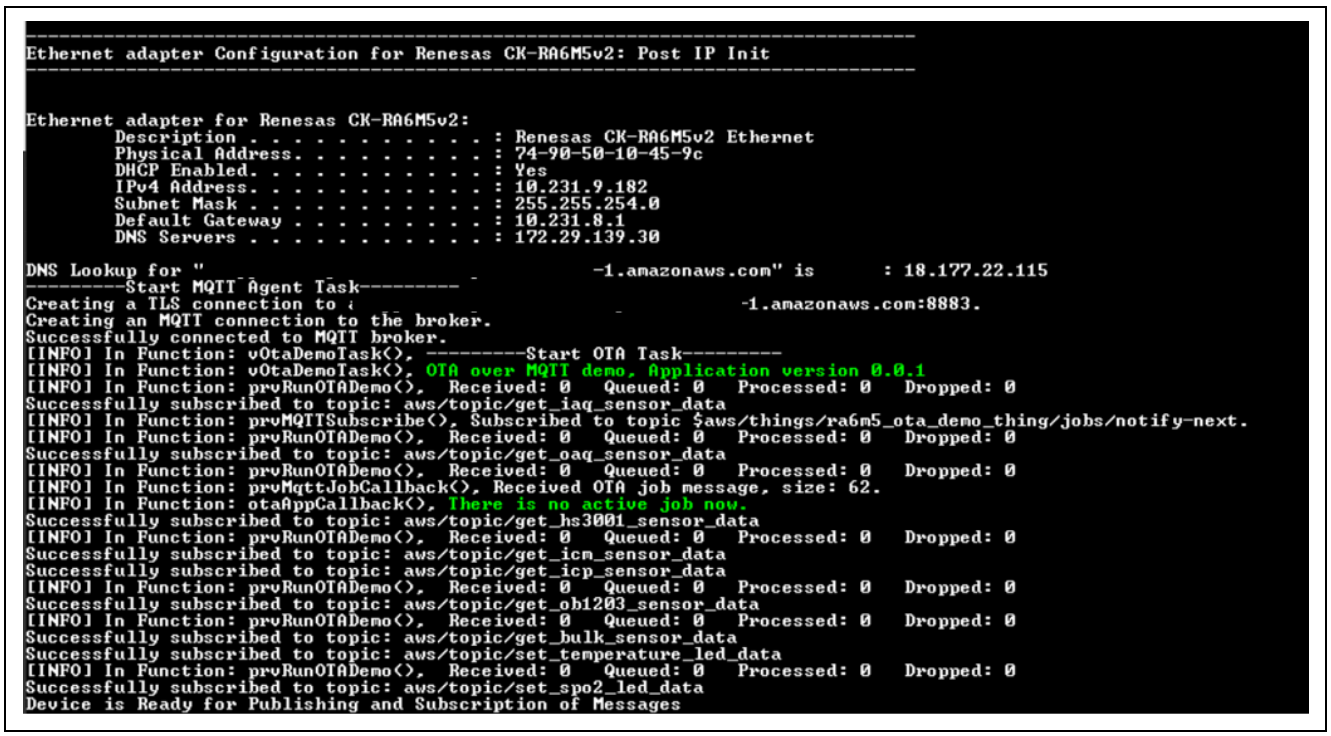

Figure 80. Connecting to the Network and AWS IoT

4.4 Verifying the Application Project from the AWS IoT

This section describes the steps for verifying this application example's functions.

Note: Wait for the board to get the IP address from the service provider upon successful Ethernet initialization and for the board to resolve the DNS lookup for the endpoint. After the successful MQTT connection message on the Console, "***Device is Ready for Publishing and Subscription of Messages***," the device is ready for Publishing and Subscribing to Messages.

For verification purposes, the user can use the AWS IoT Core Dashboard to configure and control the subscription and publish the topics as described in the following sections.

(1) Subscribe to Topics on the AWS Dashboard

On the AWS Cloud Dashboard, go to IoT Core and select **Test**, then choose **MQTT test client**. Subscribe to a topic listed below one at a time. The sample snapshot for subscribing to the topics is shown below.

```
"aws/topic/iaq_sensor_data"
"aws/topic/oaq_sensor_data"
"aws/topic/hs3001_sensor_data"
"aws/topic/icm_sensor_data"
"aws/topic/icp_sensor_data"
"aws/topic/ob1203_sensor_data"
"aws/topic/bulk_sensor_data"
```

Note: The topics shown above are **sensitive cases**; users need to take care of this while entering the publish or subscribe messages.

Only enter one topic at a time. Copy the topic 'as-is' between the quotes, and do not include any extra spaces.

Note: After the subscription to the Topics, the Dashboard is ready to receive the messages being published from the device.

Note: With the topic **"aws/topic/bulk_sensor_data",** all sensors' data will be only displayed on the AWS IoT Core Dashboard when users send a "data pull request" by publishing a message to the topic from the AWS console. This will be described in the next section (**4.4(2) Publish a Topic Messages on the AWS Dashboard**).

Note: We can subscribe to the topic "#" to monitor all messages on the AWS IoT Core.

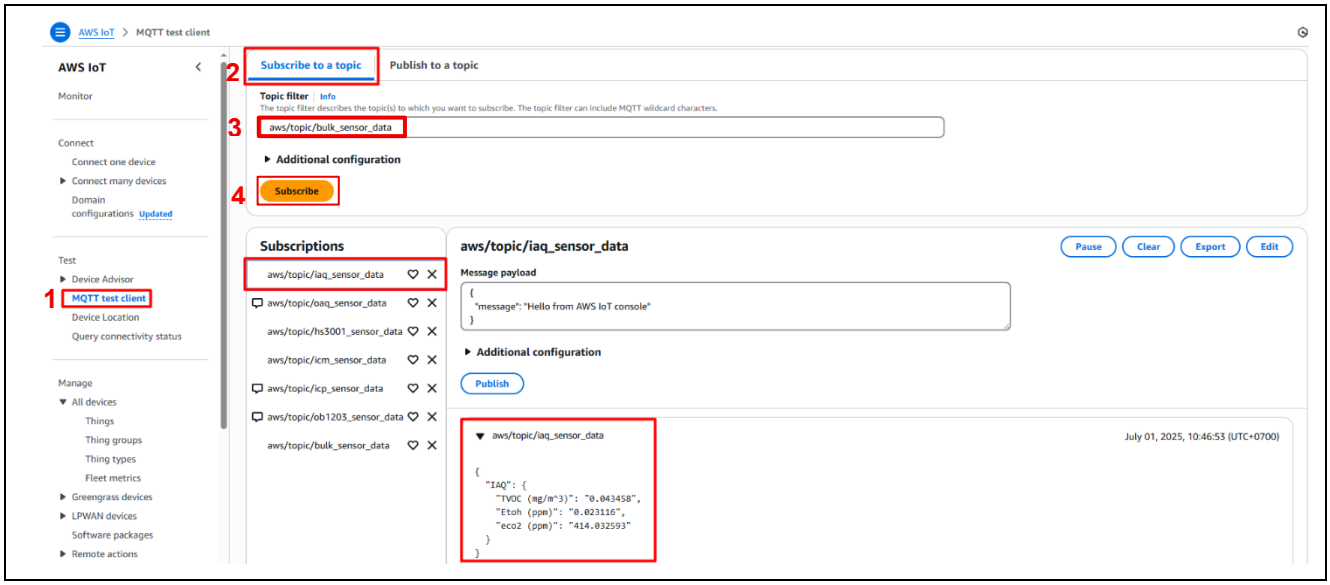

Figure 81. Click on the Subscribe button on the AWS IoT Screen to subscribe to a Topic Message

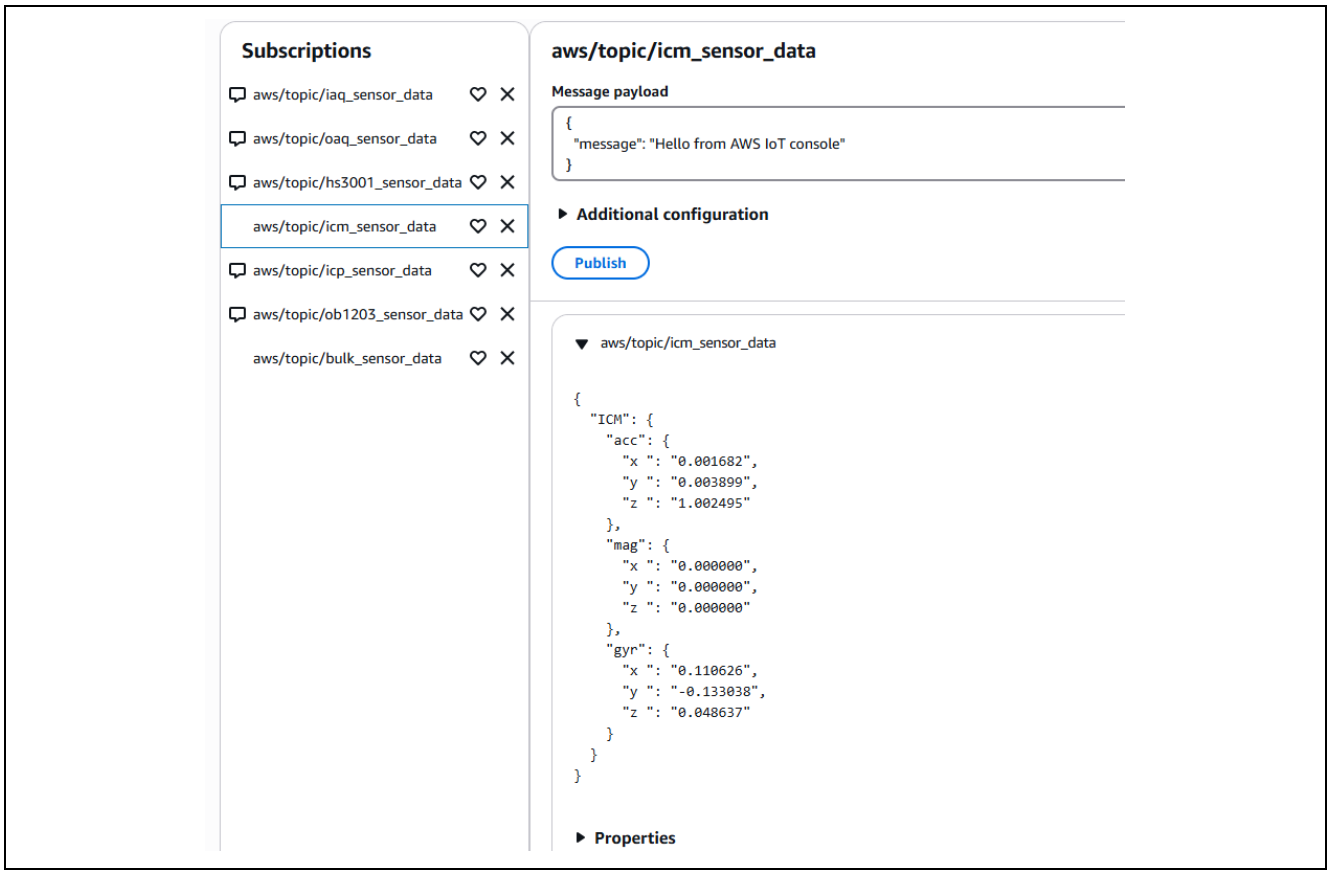

Figure 82. Subscribed Messages on the AWS IoT Screen

For more details about sensors, please refer to the section **5.2(14)** and **7**.

(2) Publish a Topic Message on the AWS Dashboard

We can publish the data below from the AWS console to verify.

HS3001 temperature alerts:

Topic: `aws/topic/set_temperature_led_data`

```
Message: {"Temperature_LED": "HOT"}     Will turn on RED in Tri-Color LED
(LED6)
Message: {"Temperature_LED": "WARM"}    Will turn on GREEN in Tri-Color LED
(LED6)
Message: {"Temperature_LED": "COLD"}    Will turn on BLUE in Tri-Color LED
(LED6)
```

Example:

Click **Test** > **MQTT test client**

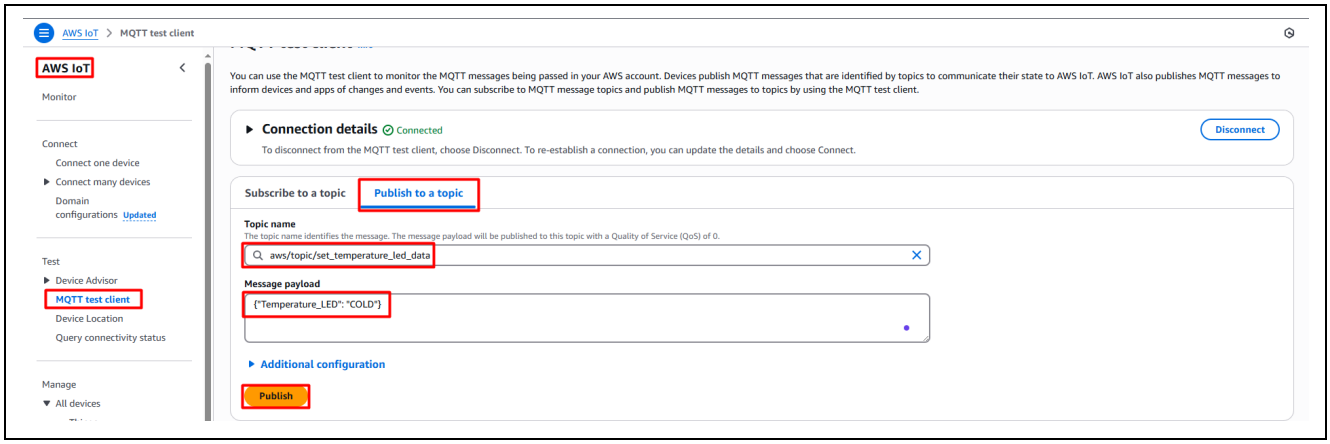

Figure 83. Publish the MQTT Message (1/3)

OB1203 SPO2 alerts:

Topic: `aws/topic/set_spo2_led_data`

```
Message: {"Spo_LED": "ON"} Will turn on BLUE LED (LED1)
Message: {"Spo_LED": "OFF"}Will turn off BLUE LED (LED1)
```

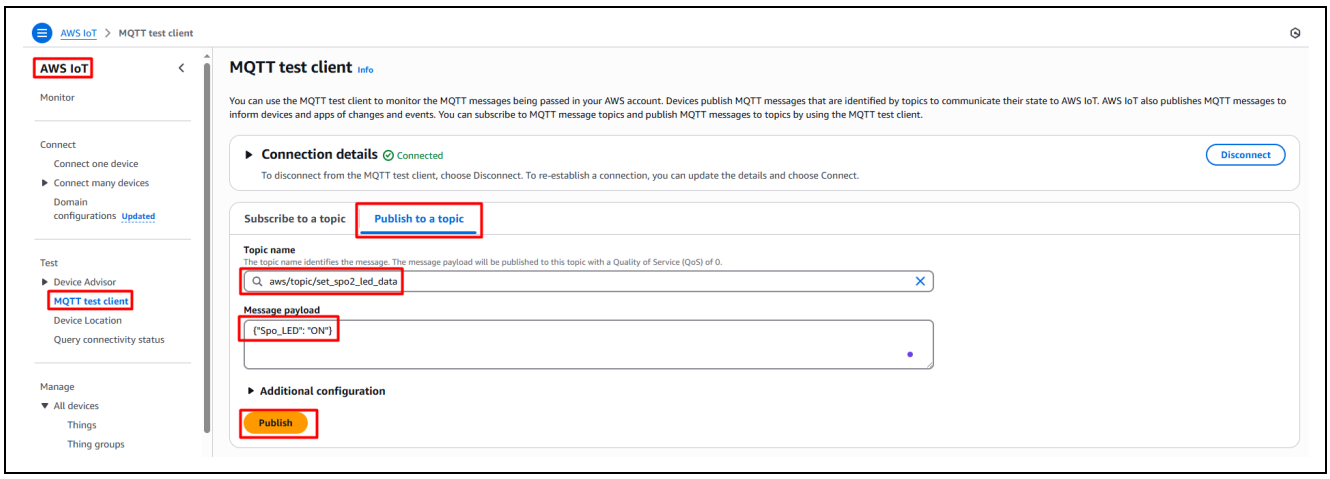

Figure 84. Publish the MQTT Message (2/3)

Data Pull Request:

Users can pull sensor values from the CK-RA6M5v2 board at their discretion by publishing messages from the AWS IoT console.

Topic:

- `aws/topic/get_iaq_sensor_data` Will get IAQ's sensor data
- `aws/topic/get_oaq_sensor_data` Will get OAQ's sensor data
- `aws/topic/get_hs3001_sensor_data` Will get HS3001's sensor data
- `aws/topic/get_icm_sensor_data` Will get ICM's sensor data

- `aws/topic/get_icp_sensor_data` `Will get ICP's sensor data`
- `aws/topic/get_ob1203_sensor_data` `Will get OB1203's sensor data`
- `aws/topic/get_bulk_sensor_data` `Will get all sensors' data from CK-RA6M5v2`

`Message: Any message`

Example: Request IAQ sensor data from AWS IoT console:

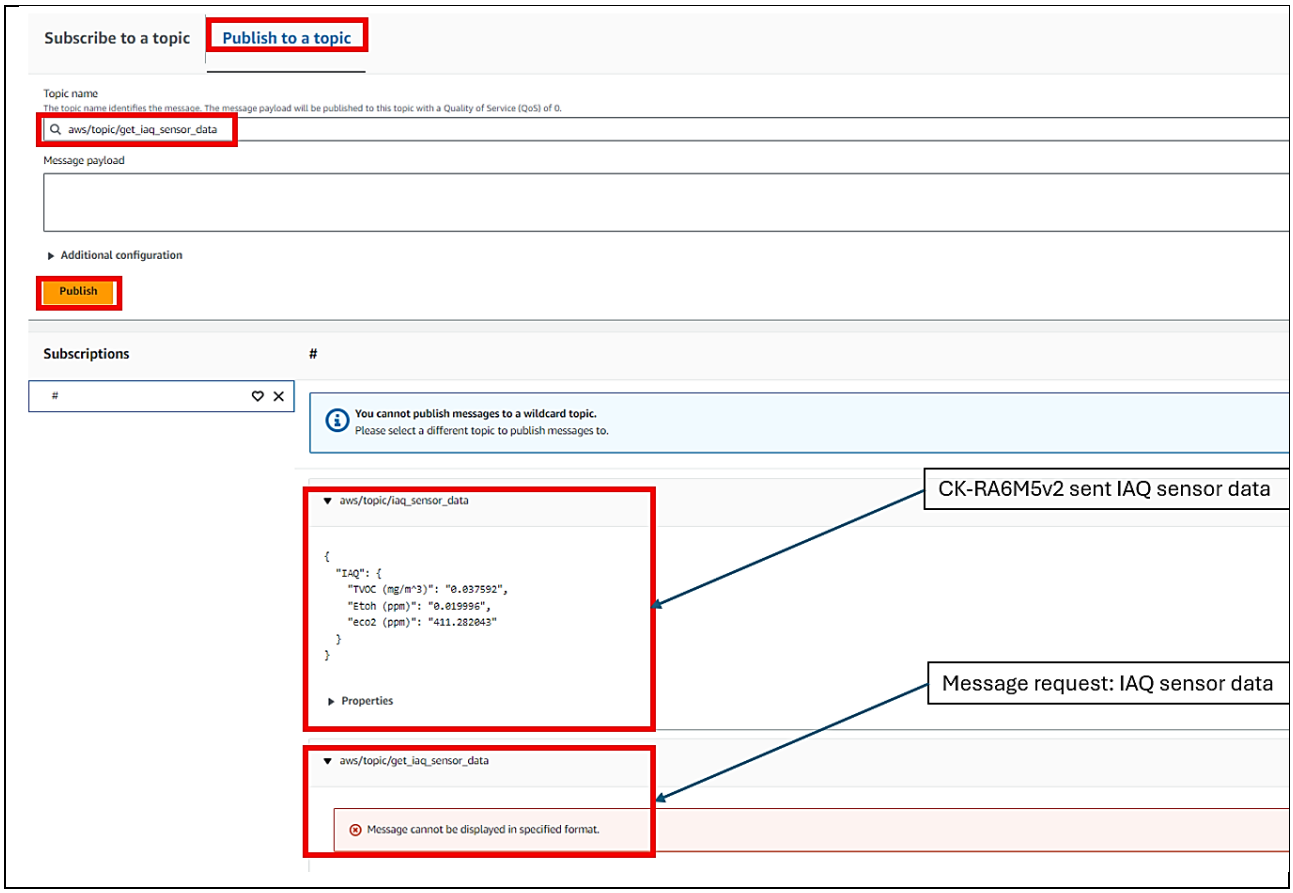

Figure 85. Publish the MQTT Message (3/3)

5. Updating the Firmware

5.1 Creating the updated firmware

This section will describe how users can create new firmware for upgrading.

The configuration for new firmware's version in **aws_ck_ra6m5_v2_ethernet_ota_app > configuration.xml > Stacks > AWS IoT Over-the-air Update Library > Open Properties > Common > Version >**

- Major
- Manor
- Build

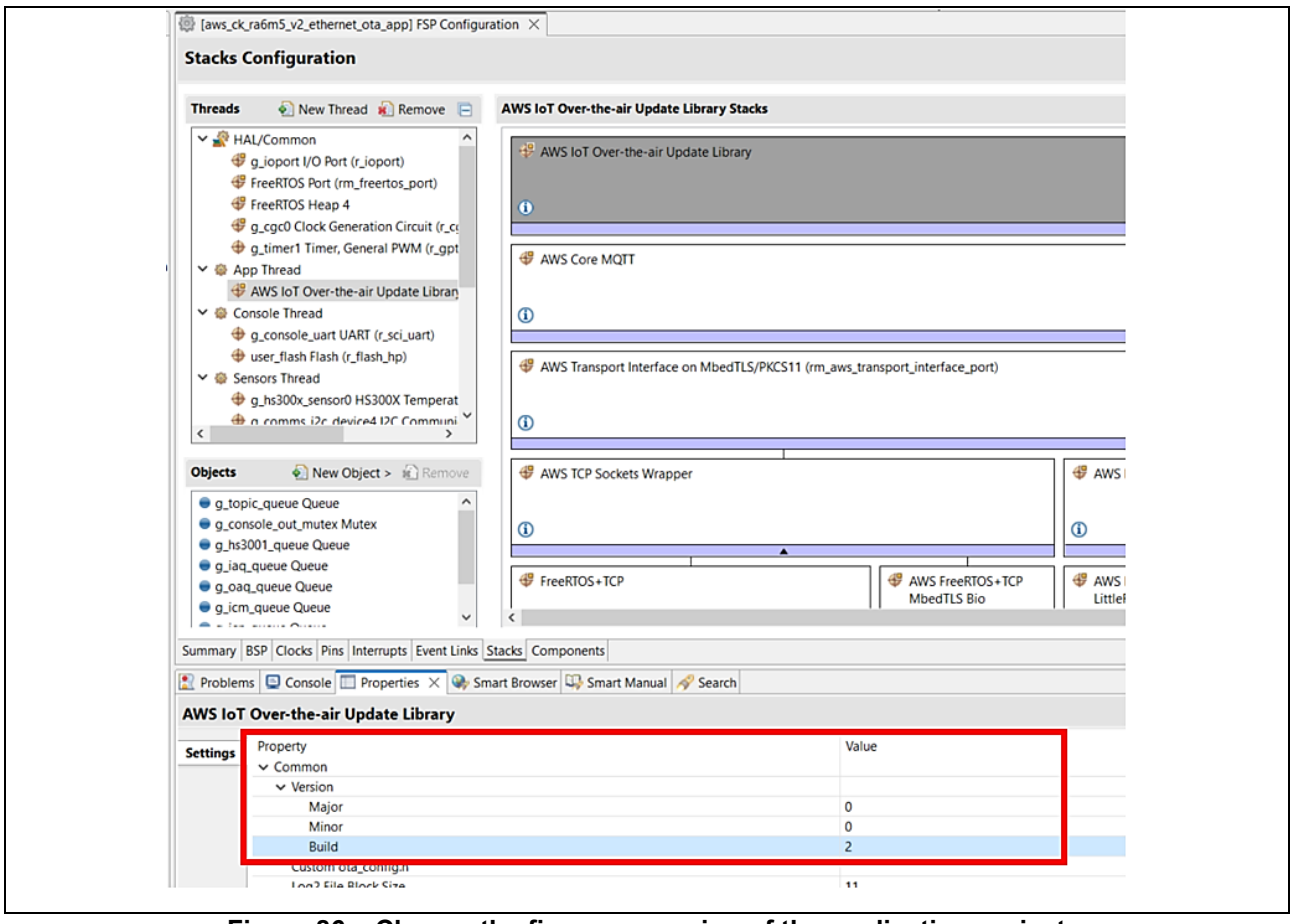

Figure 86. Change the firmware version of the application project

The OTA application will use this version value to compare with the previous version, thereby determining whether a firmware upgrade is allowed or not. Then it will publish the result message to AWS IoT Core.

Note: In the default setting, only the firmware, which has a version higher than the previous firmware, can be updated. And in this case, the “Allow Downgrade” option has a “Disallowed” value.

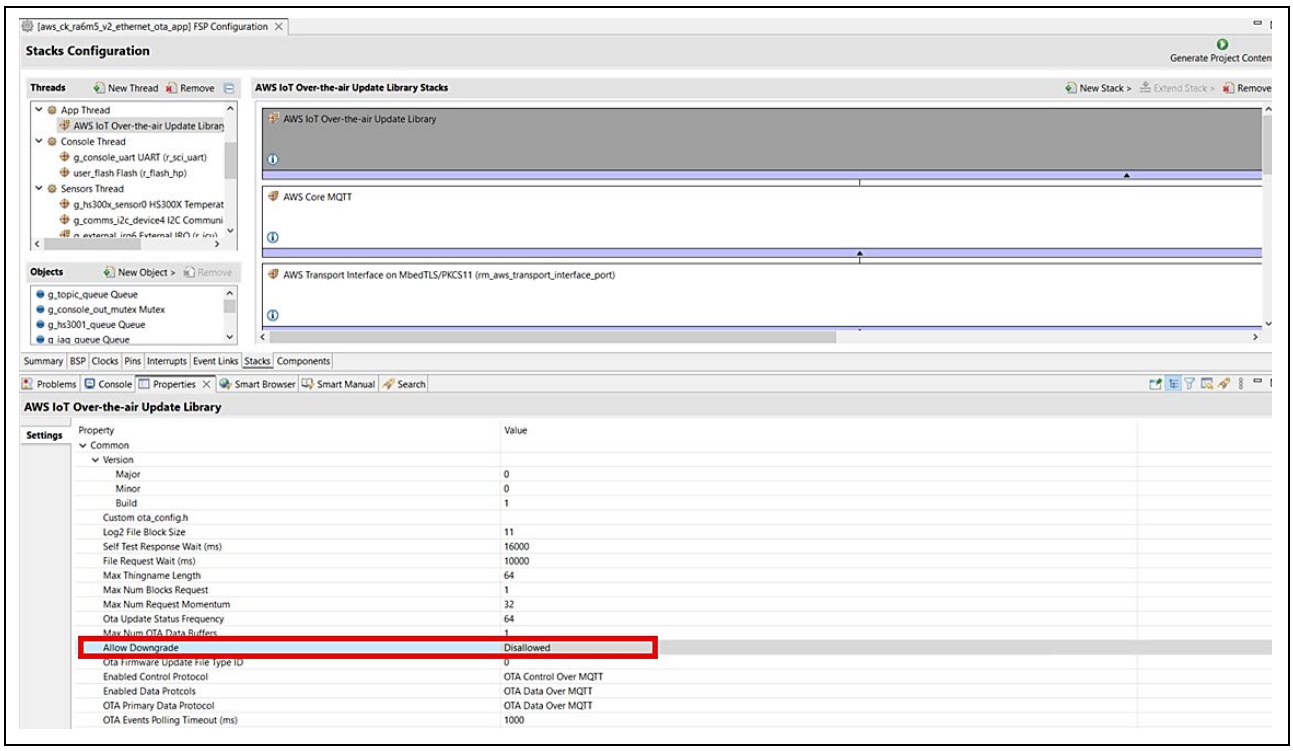

Figure 87. "Allow Downgrade" feature

Users can enable this option to utilize the "Allow Downgrade" feature. This feature allows users to upgrade to new firmware without needing to compare the version.

After the configuration is finished, please refer to the section 3.3 to build and generate the new signed image at `\Debug` folder:
`aws_ck_ra6m5_v2_ethernet_ota_app\Debug\aws_ck_ra6m5_v2_ethernet_ota_app.bin.signed`. This file will be used to upgrade in the next section, **5.2(10) Update the firmware.**

Note: After creating new firmware, if users are still using debug on e² studio, it will prompt users to confirm reloading a new image onto the board. In this case, please deny reloading the new image to the board by clicking "**Remember my decision**" and choosing "**Continue**":

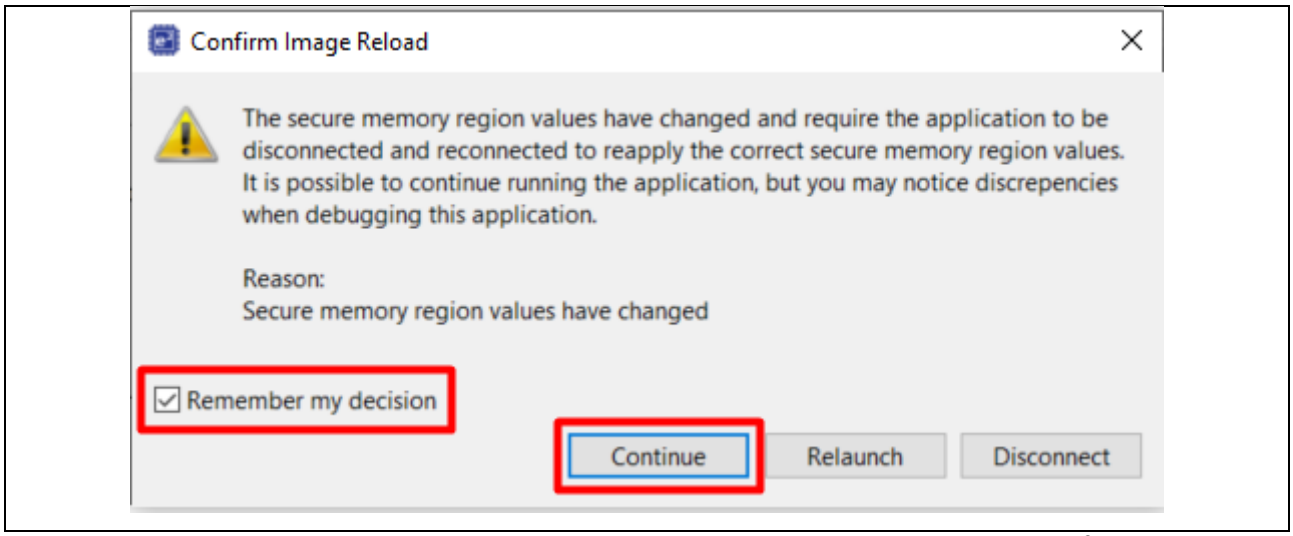

Figure 88. Refuse to reload the new image after creating new firmware via e² studio

5.2 Updating the firmware

In AWS, create an OTA update job that will update the firmware.

(1) In the IOT Core menu, select **Manage**, **Remote actions**, and **Jobs**, and then click the **Create job** button

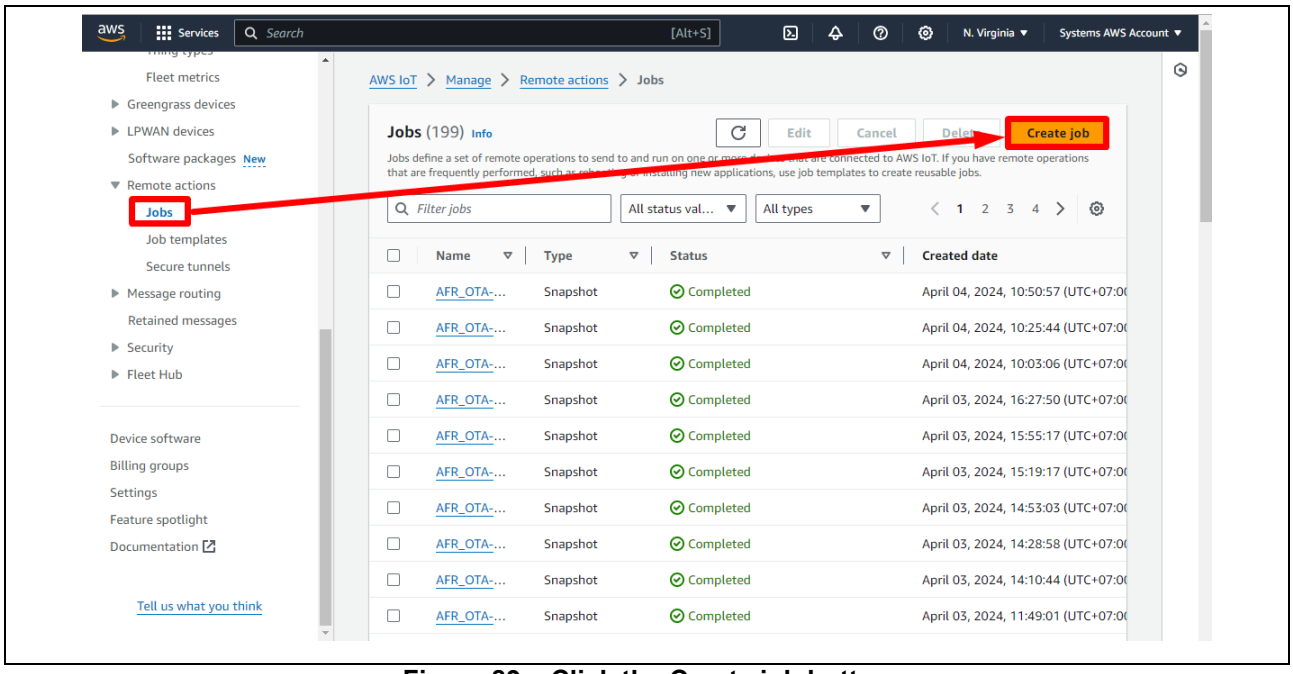

Figure 89. Click the Create job button

(2) Select **Create FreeRTOS OTA update job** and then click **Next**

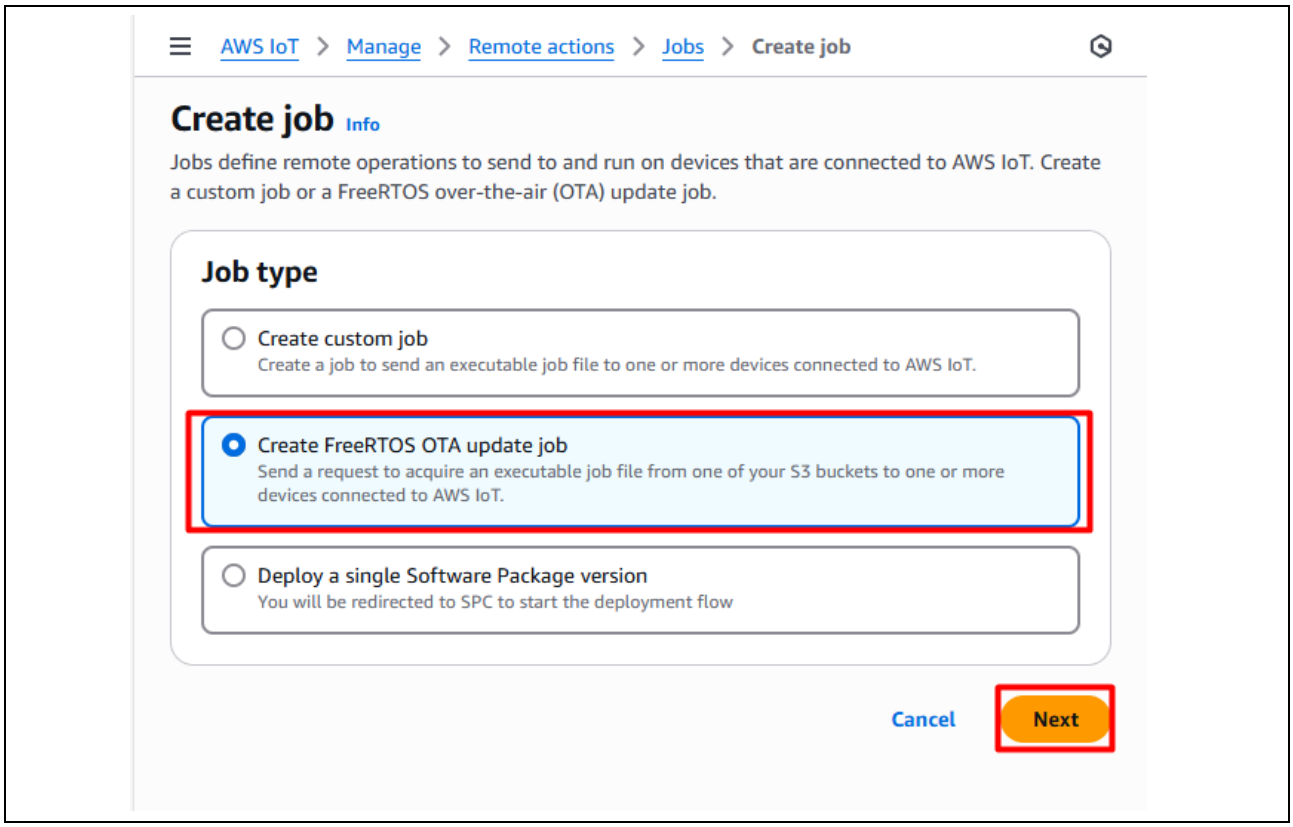

Figure 90. Select the Create FreeRTOS OTA update job

(3) Enter a job name (example: ra6m5_ota_demo_job) and then click **Next**

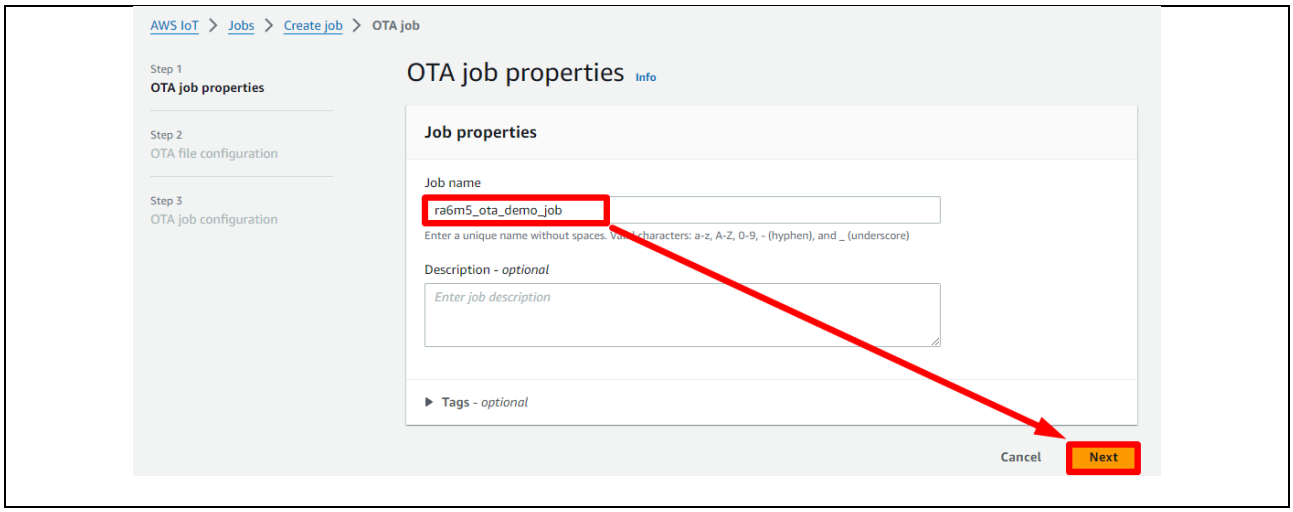

Figure 91. Enter an OTA job name

(4) Click the **Devices to update** drop-down list and select the device to update

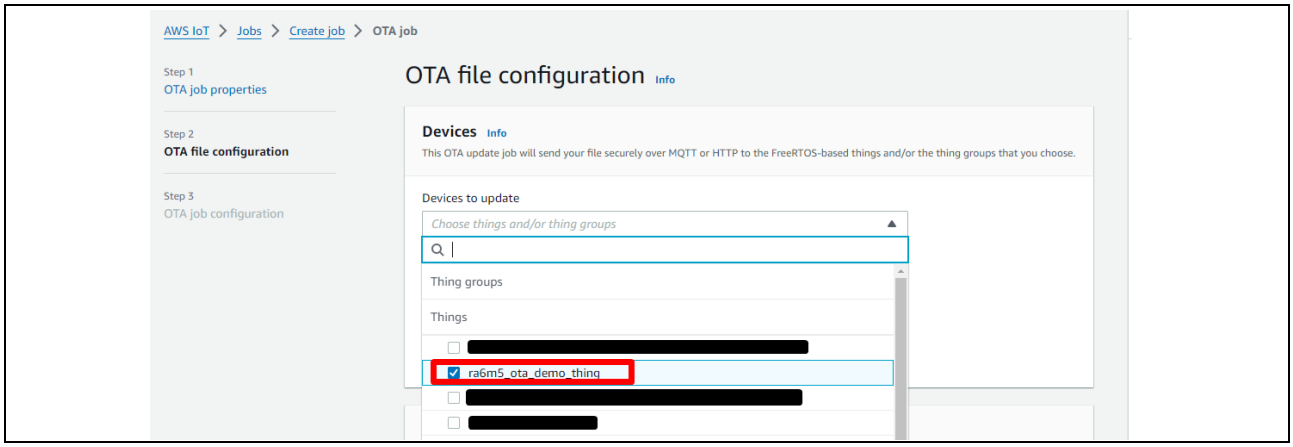

Figure 92. Select the device to update

(5) Click **Create new profile**

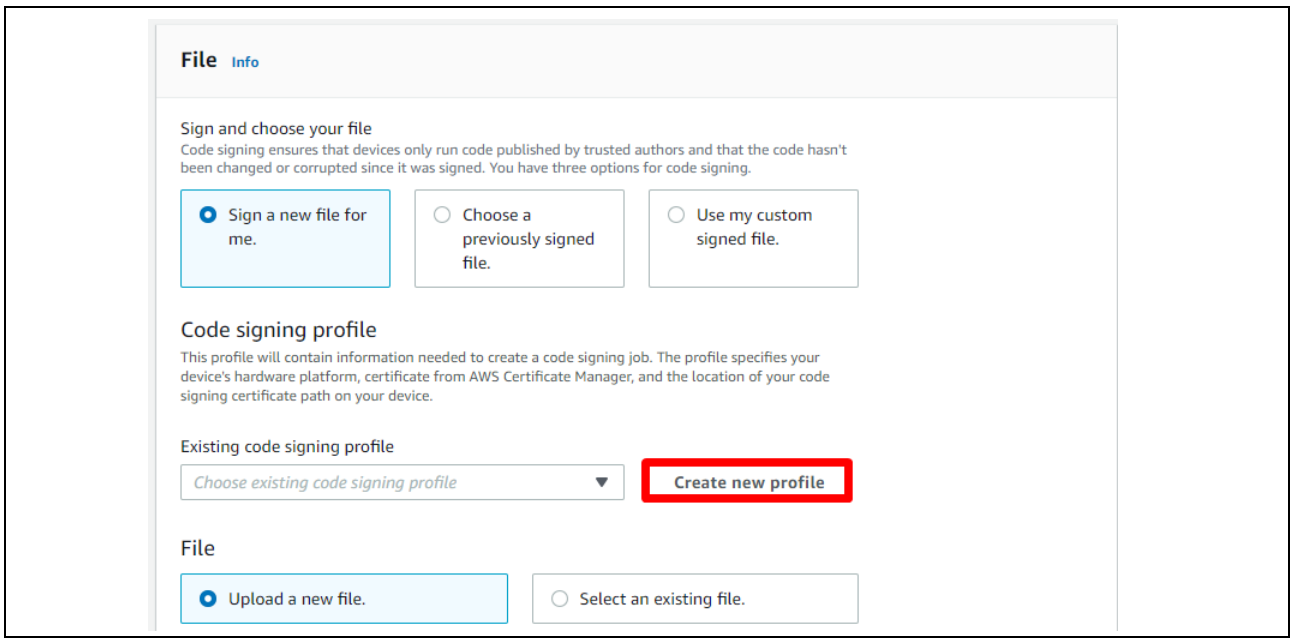

Figure 93. Click the Create new profile button

You can skip steps (5) to (9) if you have already created a profile. Click **Choose existing code signing profile** and select the profile you created from the drop-down list.

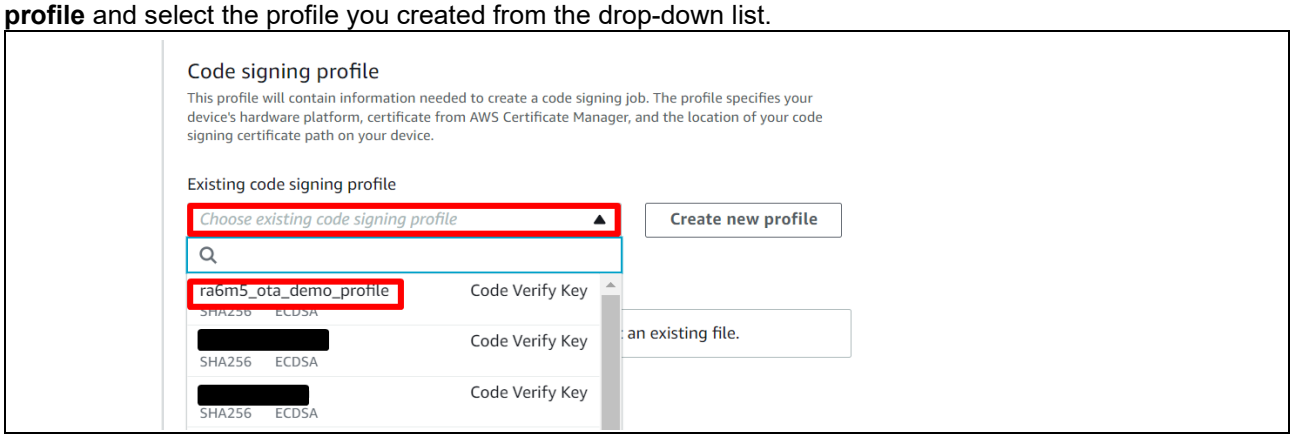

Figure 94. Choose an existing code signing profile

(6) Create a profile (1): Profile name and device hardware platform

- Enter the profile name (example: ra6m5_ota_demo_profile)
- Select **Windows Simulator** as the device hardware platform

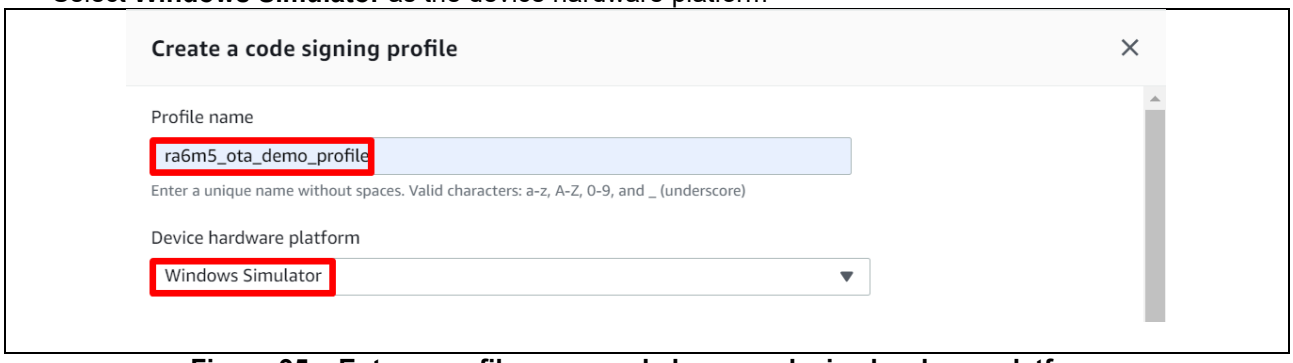

Figure 95. Enter a profile name and choose a device hardware platform

(7) Create a profile (2): Import a certificate

- In the **Code signing certificate** area, click **Import new code signing certificate**
- In the **Certificate body**, select the file `secp256r1.crt` you created in section **3.2.1(6)**
- In the **Certificate private key**, select the file `secp256r1.privatekey` you created in section **3.2.1(7)**
- In the **Certificate chain**, select the file `ca.crt` you created in section **3.2.1(3)**
- Click **Import**

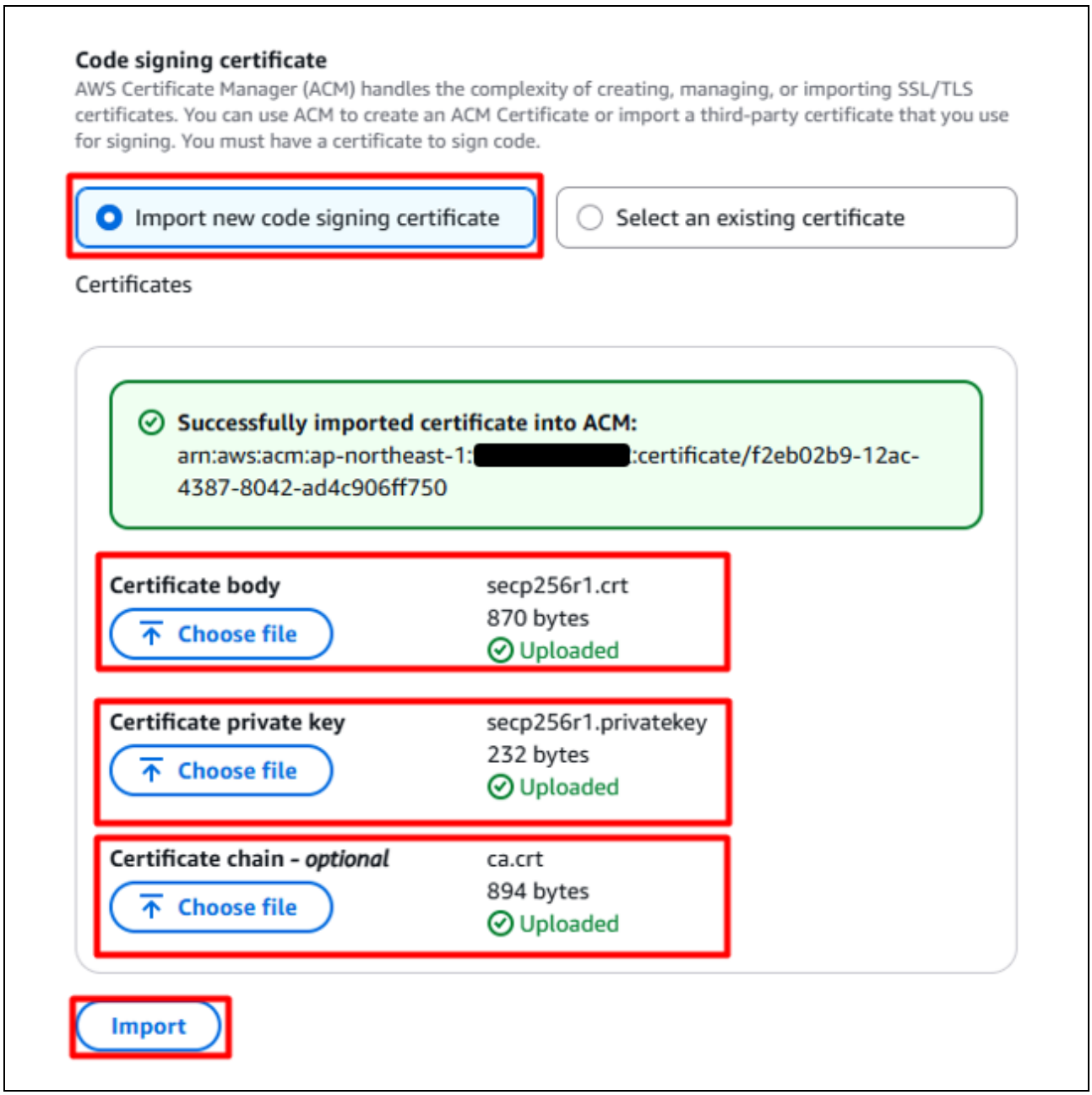

Figure 96. Create a code signing profile

(8) Create a profile (3): Enter the path of the code signing certificate of the device: **"Code Verify Key"** and then click **Create**

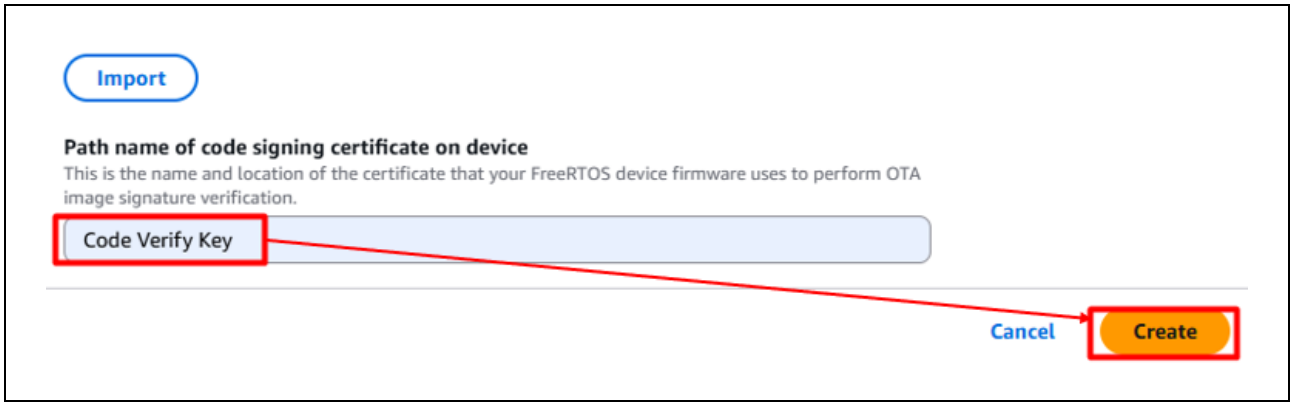

Figure 97. Enter the path of the code signing certificate of the device

(9) Confirm that the name of the profile you created earlier is selected in the **Existing code signing profile** drop-down list

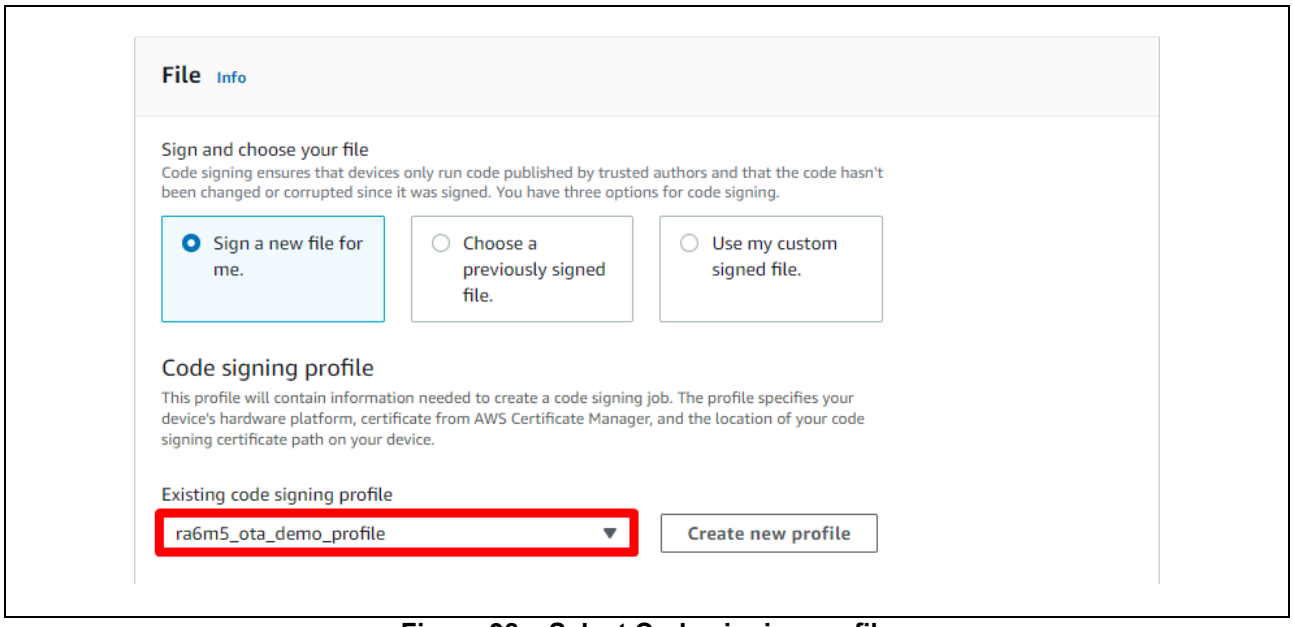

Figure 98. Select Code signing profile

(10) Update the firmware

- Select **Upload a new file**
- In **File to upload**, select the file `aws_ck_ra6m5_v2_ethernet_ota_app.bin.signed` you built in the folder `aws_ck_ra6m5_v2_ethernet_ota_app\Debug` in section **5.1.**
- Click **Browse S3** and select the S3 bucket you created in section **2.1.3.4**.
- Enter a path name in **Path name of file on device** (You can enter any path name. Example: /device/updates)

Figure 99. Upload new file

(11) In the **Role** drop-down list, select the role you created in section **2.1.3.5** and then click **Next**

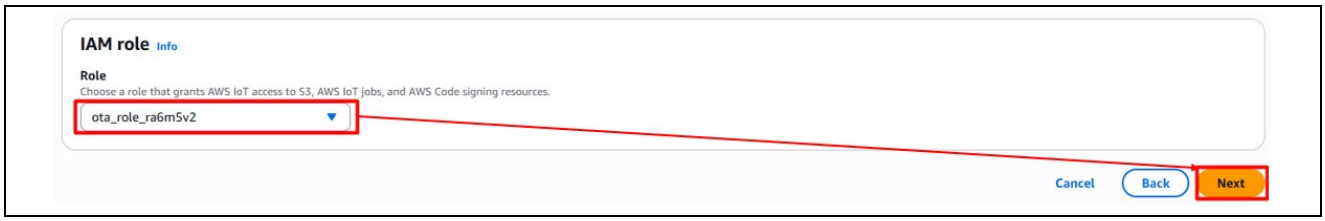

Figure 100. Choose IAM role

(12) Click **Create job**

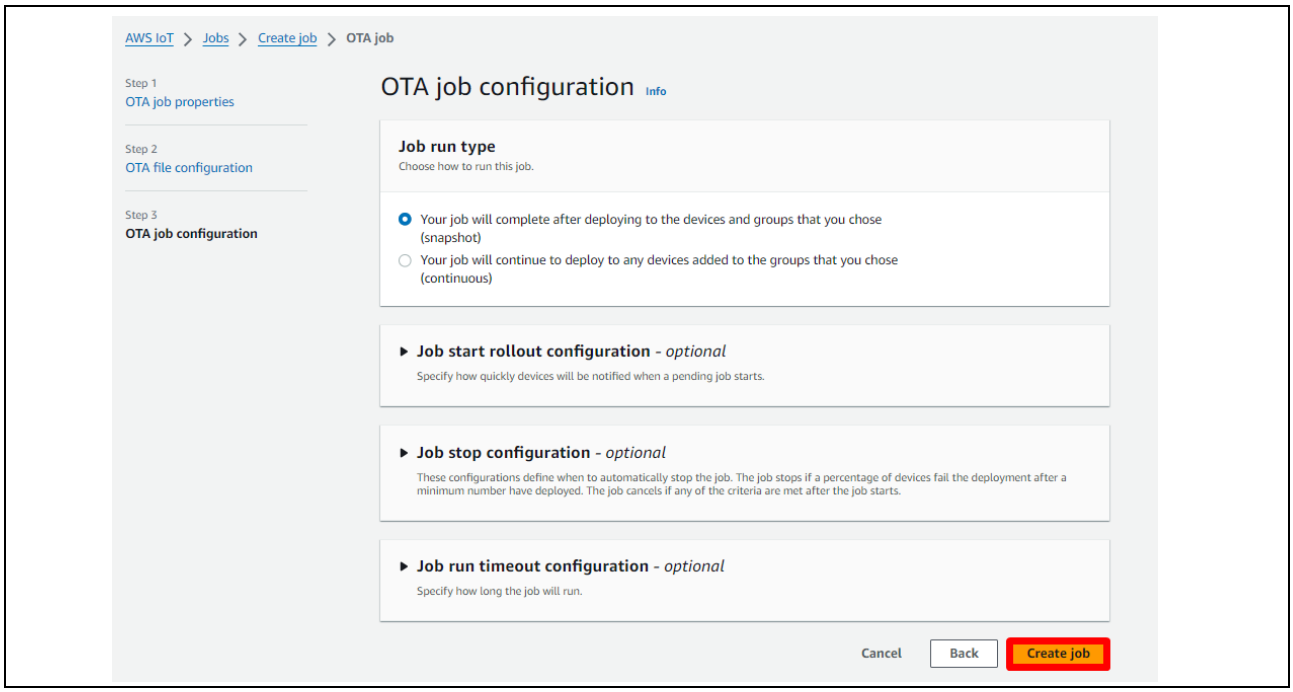

Figure 101. Click the Create job button

(13) Wait until firmware reception is complete

When the job starts, the job receives and writes the firmware.

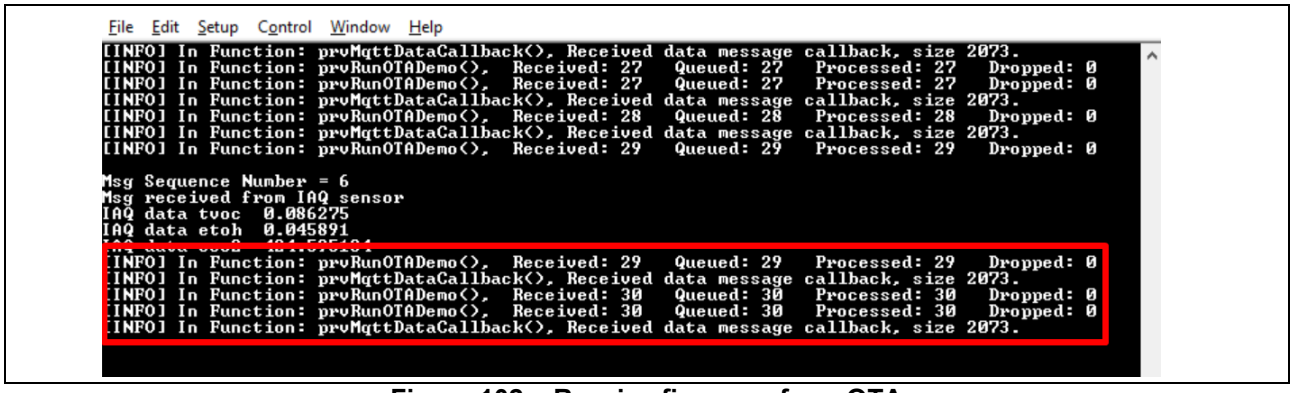

Figure 102. Receive firmware from OTA

When the update process is complete, the device will reset, and the initial menu will appear.

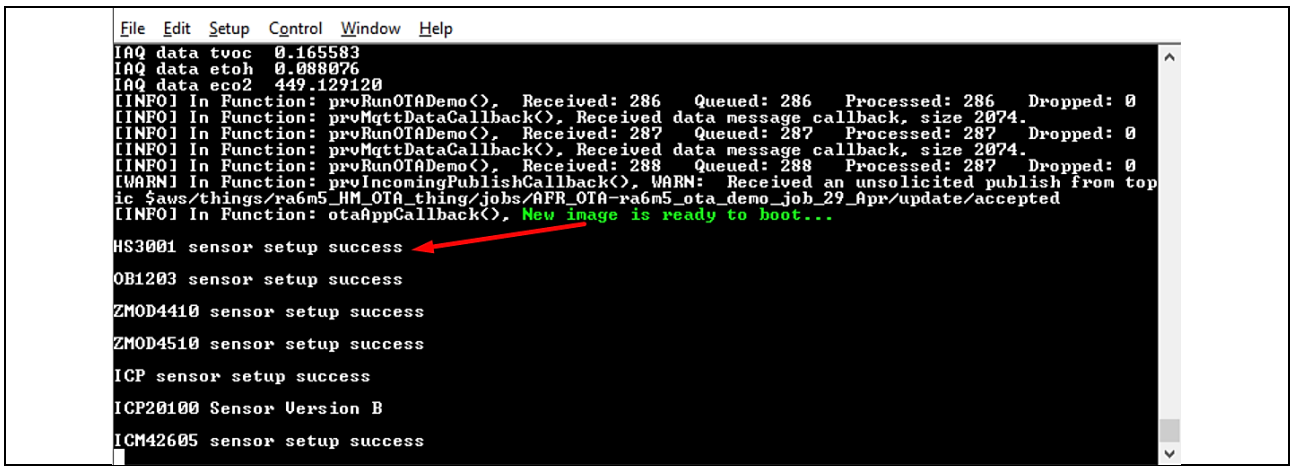

Figure 103. Active new firmware image

(14) Confirm updating result

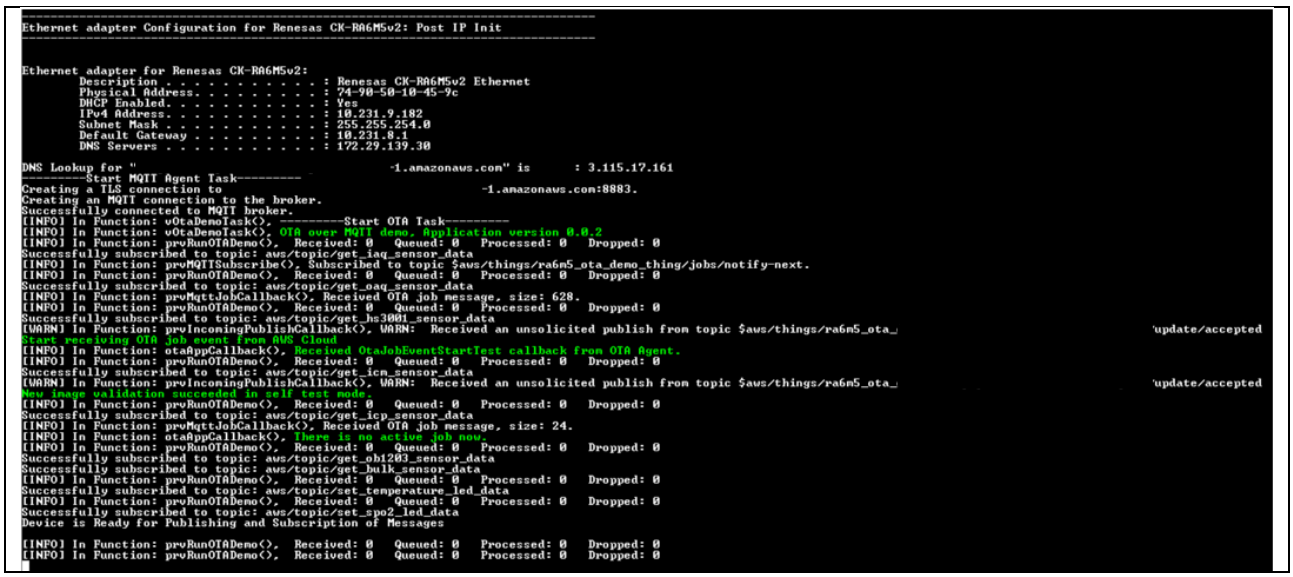

Figure 104. Version of the new firmware image and validation

Go to the AWS IoT Core console, select **Manage**, **Remote actions**, and **Jobs,** and choose the job that was created in the previous step.

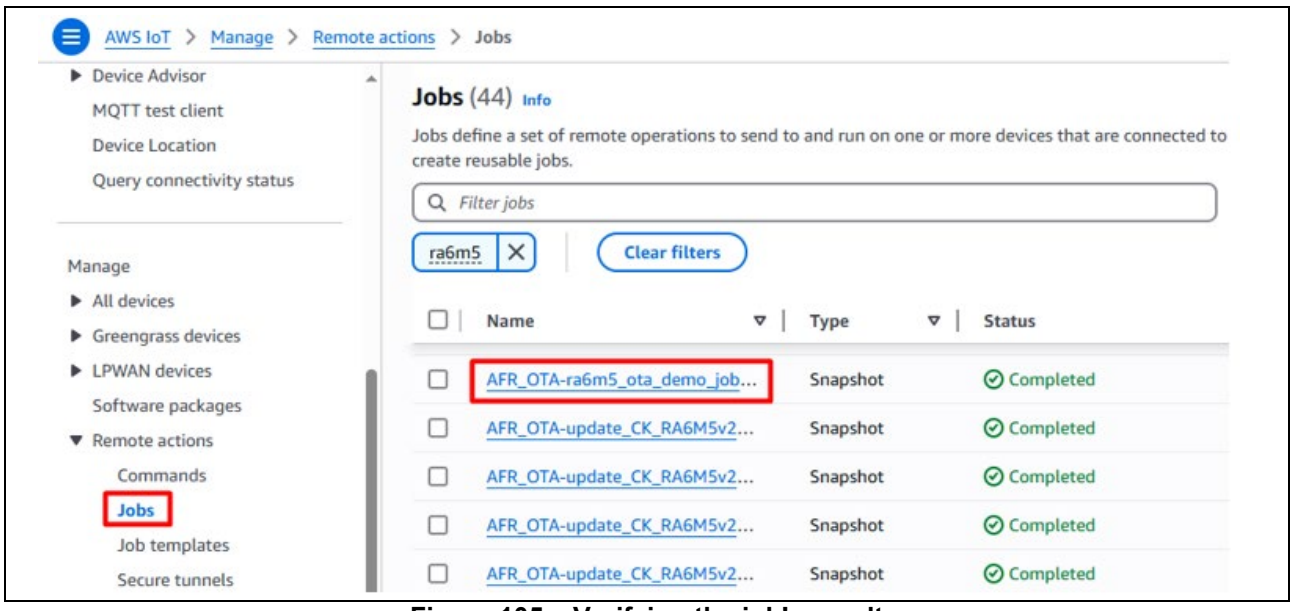

Figure 105. Verifying the job's result

Choose “Job executions” to check the current result of a job.

- Succeeded (upgrading firmware successfully):

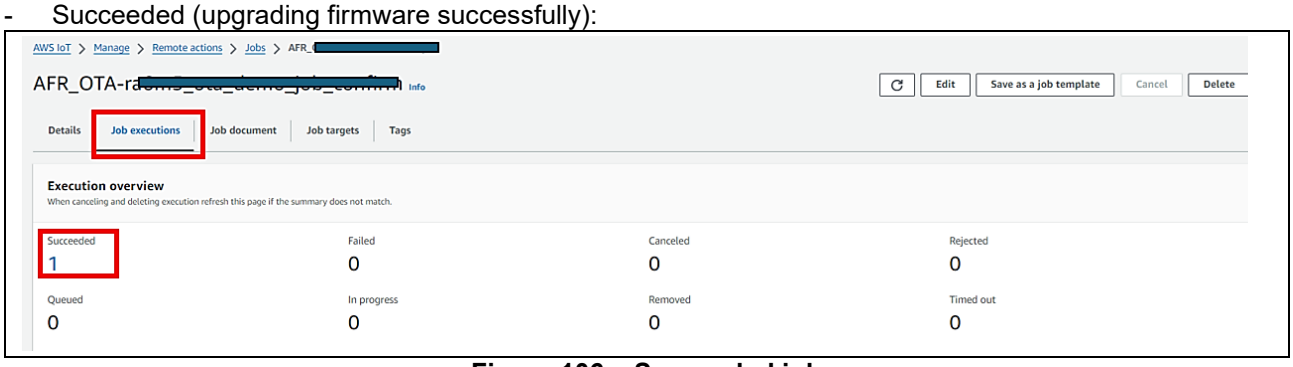

Figure 106. Succeeded job

In this case, the firmware, which has been upgraded, has a higher version than the version of the old firmware. Or users used the “Allow downgrade” feature. (Please refer to the section **5.1** for more information).

- Failed (upgrading firmware unsuccessfully):

Figure 107. Failed job

In this case, the firmware that users chose to upgrade has a version equal to or smaller than the old firmware’s version. For the AWS code signing that users set up for the device, the OTA job is incorrect. Users need to check section **3.2, 4.2** (add Code Signing Public Key)**, 5.2(7)** (import certificate) again.

6. Sensor Data for Cloud Kits

The AWS dashboard displays the following Data from sensors.

Table 1. Sensor Data in AWS IoT Console

Sensor	Data
HS3001- Humidity and Temperature Sensor	Temperature, F
	Humidity, %
ZMOD4410- Indoor Air Quality Sensor	Etoh, ppm
	ECO2- Estimated Carbon dioxide, ppm
	TVOC - Total Volatile Organic Compounds, mg/m^3
OB1203 - Heart Rate, Blood Oxygen Concentration, Pulse Oximetry, Proximity, Light, and Color Sensor	SPO2, %
	HR (Heart Rate), bpm (beats per minute)
	RR (Respiration Rate), breaths per minute
	P2P
ICP-20100 - Barometric Pressure and Temperature Sensor	Temperature, F
	Barometric Pressure, mbar
ICM-42605 Motion Tracking Sensor	Acc values, unit: g
	Gyro Data, unit: dps (degrees per sec)
	Mag Data, unit: mT
ZMOD4510 – Outdoor Air Quality	OAQ, ppm

7. Sensor Stabilization Time

Table 2 gives the time required for the sensors to sense and provide valid data to the users. Here, you will see 2 columns: 1) when powered up for the first time, and 2) after a soft or hard reset. If the system boots up from a cold start, the time for the sensors to provide the valid data is up to (1 minute – 4 hours), whereas if the system boots up from a warm start, the time for the sensors to provide the valid data is up to (10 seconds – 2 hours). For more details, refer to the specific sensor datasheet.

Table 2. Sensor Stabilization Time

Sensor Name	When Powered Up for the First Time	After Soft or Hard Reset
ZMOD4410 IAQ	Up to 1 minute	Up to 1 minute
ZMOD4510 OAQ	Up to 4 hours	Up to 5 minutes
OB1203	Up to 20 minutes (After placing a finger on the sensor, it may take up to 60 seconds to sense data)	Up to 20 seconds (After placing a finger on the sensor, it may take up to 60 seconds to sense data)
HS3001	Up to 1 minute	Up to 10 seconds
ICP	Up to 1 minute	Up to 10 seconds
ICM	Up to 1 minute	Up to 10 seconds

Note: The stabilization time of the sensor provided above is from the point of sensor initialization.

8. Known Issues and Troubleshooting.

- This section talks about the known FSP and tool-related issues. More details can be found at the link: https://github.com/renesas/fsp/issues.
- When running debug on the e^2 studio, if the application is rerun multiple times, it might randomly cause an issue with the I2C communication of the OB1203 sensor. Users need to reconnect the micro-USB cable (J10) and USB-C cable (J28) to cold reset the OB1203 sensor and run the application again.
- When firmware reception from the OTA job finishes, but the new firmware does not start, users need to check that the firmware update was successful from console logs. Also, revisit the steps to make sure AWS code signing and certificate generation are written to the device flash (3.2, 4.2), creating the code signing profile step (5.2(7)), or the path of code signing certificate of the device (5.2(8)) was properly followed and successful.

9. Debugging

Enable the `USR_LOG_LVL (LOG_DEBUG)` macro in the application project for additional information on the error during debugging.

10. References

[1] International Telecommunication Union, "ITU-T Y.4000/Y.2060 (06/2012)," 15 06 2012. [Online]. Available: http://handle.itu.int/11.1002/1000/11559.

[2] Amazon Web Services, "AWS IoT Core Features," [Online]. Available: https://www.amazonaws.cn/en/iot-core/features/.

[3] Amazon Web Services, "AWS IoT Core," [Online]. Available: https://www.amazonaws.cn/en/iot-core/.

[4] W. T. L. L. O. S. R. N. S. R. X. G. K. N. K. S. F. M. K. D. L. I. R. Valerie Lampkin, Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry, IBM Redbooks, 2012.

[5] I. E. T. Force, "The Transport Layer Security (TLS) Protocol Version 1.2," [Online]. Available: https://tools.ietf.org/html/rfc5246.

[6] Amazon Web Services, "AWS IoT Security," [Online]. Available: https://docs.aws.amazon.com/iot/latest/developerguide/iot-security.html.

[7] Amazon Web Services, "Transport Security in AWS IoT," [Online]. Available: https://docs.aws.amazon.com/iot/latest/developerguide/transport-security.html.

[8] International Telecommunication Union, "X.509 (10/19) Summary," 10 2019. [Online]. Available: https://www.itu.int/dms_pubrec/itu-t/rec/x/T-REC-X.509-201910-I!!SUM-HTM-E.htm.

[9] Eclipse Foundation, "Eclipse Mosquitto™ - An open source MQTT broker," [Online]. Available: https://mosquitto.org/.

[10] Amazon Web Services, "AWS IoT Device SDK C: MQTT," [Online]. Available: https://docs.aws.amazon.com/freertos/latest/lib-ref/c-sdk/mqtt/index.html.

[11] R. Barry, "Mastering the FreeRTOS™ Real Time Kernel," in *A Hands-On Tutorial Guide*, 2016.

[12] A. I. D. S. C. Documentation, "AWS IoT Device SDK C: MQTT Functions," [Online]. Available: https://docs.aws.amazon.com/freertos/latest/lib-ref/c-sdk/mqtt/mqtt_functions.html.

[13] Amazon, "Configuring the FreeRTOS Demos," [Online]. Available: https://docs.aws.amazon.com/freertos/latest/userguide/freertos-configure.html.

[14] "Amazon FreeRTOS mbedTLS," [Online]. Available: https://github.com/aws/amazon-freertos/blob/master/libraries/3rdparty/mbedtls/utils/mbedtls_utils.c.

[15] Renesas Electronics Corporation, "Renesas Flash Programmer (Programming GUI) - Documentation," [Online]. Available: https://www.renesas.com/us/en/products/software-tools/tools/programmer/renesas-flash-programmer-programming-gui.html#documents.

[16] Amazon Web Services, "FreeRTOS Over-the-Air Updates," [Online]. Available: https://docs.aws.amazon.com/freertos/latest/userguide/freertos-ota-dev.html.

[17] Nordic Semiconductor, "Bootloader," [Online]. Available: https://developer.nordicsemi.com/nRF_Connect_SDK/doc/latest/mcuboot/design.html.

[18] Renesas Electronics Corporation, "AWS OTA PAL on MCUBoot," [Online]. Available: https://renesas.github.io/fsp/group___a_w_s___o_t_a___p_a_l___m_c_u_b_o_o_t.html.

[19] Renesas Electronics Corporation, "Booting Encrypted Image using MCUboot and QSPI," [Online]. Available: https://www.renesas.cn/cn/zh/document/apn/booting-encrypted-image-using-mcuboot-and-qspi-application-project.

[20] Renesas Electronics Corporation, "How to implement FreeRTOS OTA using Amazon Web Services in RX65N," [Online]. Available: https://www.renesas.com/us/en/document/apn/rx-family-how-implement-freertos-ota-using-amazon-web-services-rx65n-v20221001-lts-rx-110-or-later.

[21] Renesas Electronics Corporation, "RA6 Basic Secure Bootloader Using MCUboot and Internal Code Flash," [Online]. Available: https://www.renesas.com/us/en/document/apn/ra6-basic-secure-bootloader-using-mcuboot-and-internal-code-flash.

Website and Support

Visit the following vanity URLs to learn about key elements of the RA family, download components and related documentation, and get support.

CK-RA6M5v2 Kit Information	renesas.com/ra/ck-ra6m5
RA Cloud Solutions	renesas.com/cloudsolutions
RA Product Information	renesas.com/ra
RA Product Support Forum	renesas.com/ra/forum
RA Flexible Software Package	renesas.com/FSP
Renesas Support	renesas.com/support

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Aug.22.24	—	Initial release
1.10	Oct.01.25	—	Migrated to FSP 6.1.0

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)

   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on

   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state

   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins

   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin

   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.