NXP Semiconductors
User's Guide

Document identifier: AWSOTAUG
Rev. 1, 15 June 2020

Amazon FreeRTOS Over-The-Air Updates using i.MX RT1060

Contents

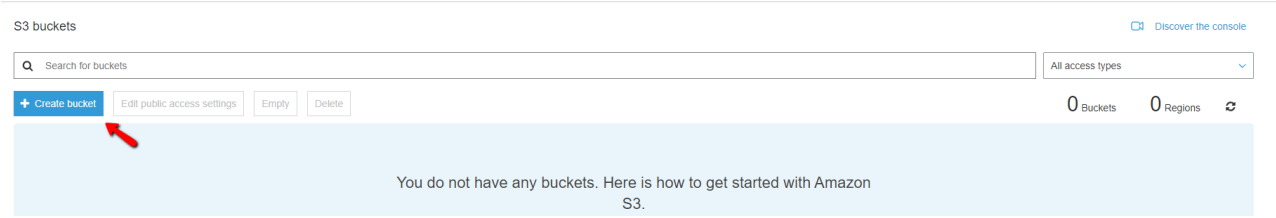
Chapter 1
Overview

This guide walks through the steps to configure AWS services to make an Amazon FreeRTOS Over The Air Update using NXPs RT1060-EVK SDK. First, it creates an IAM role with OTA update, S3, IoT policies, and permissions. Then, using OpenSSL and AWS CLI commands, a code signing certificate is issued. Finally, it shows how to create an IoT thing with the code signing certificate with an OTA job.

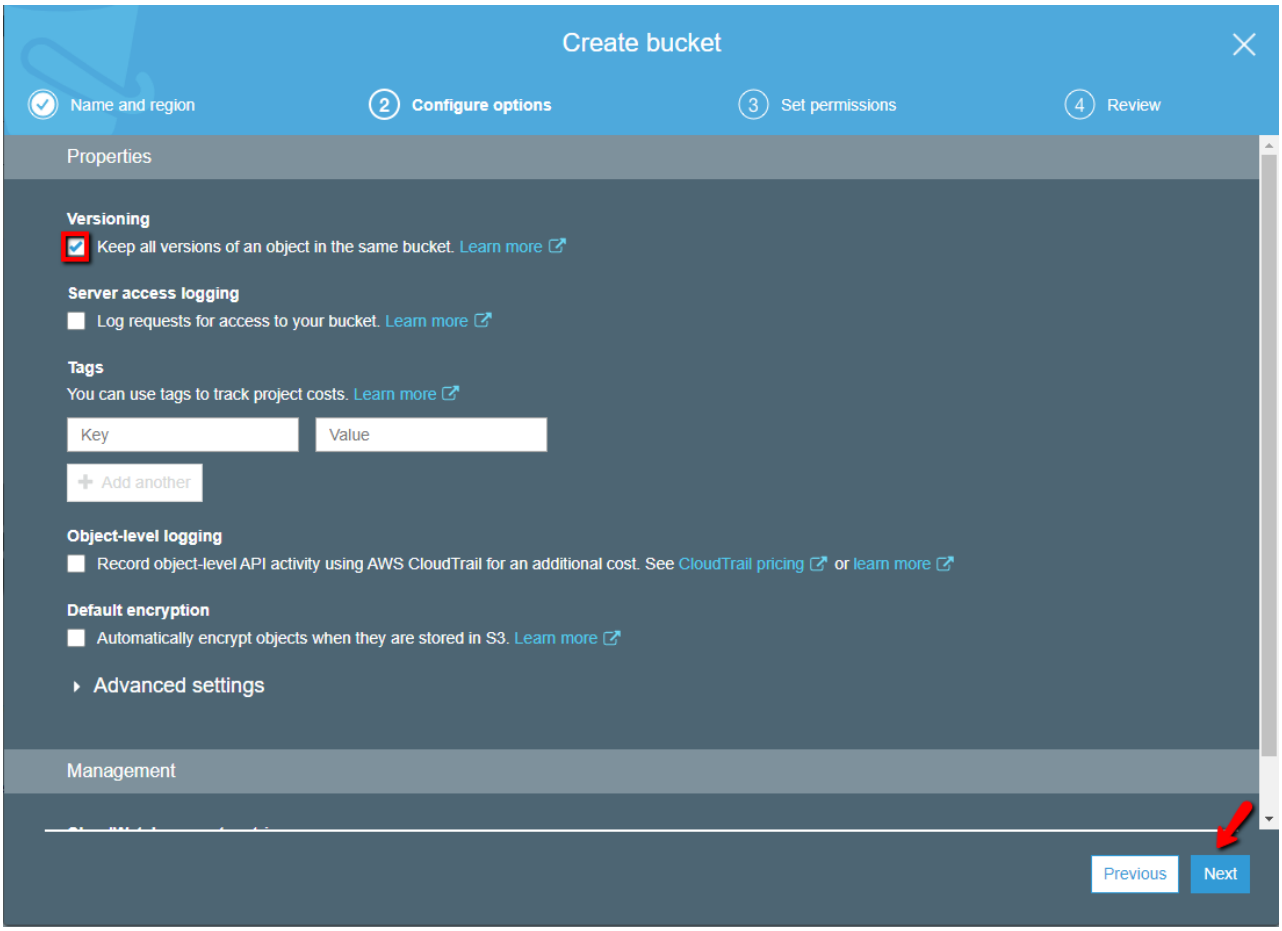
Chapter 2
AWS OTA Pre-Requisites

2.1 Create an Amazon S3 Bucket to store your update

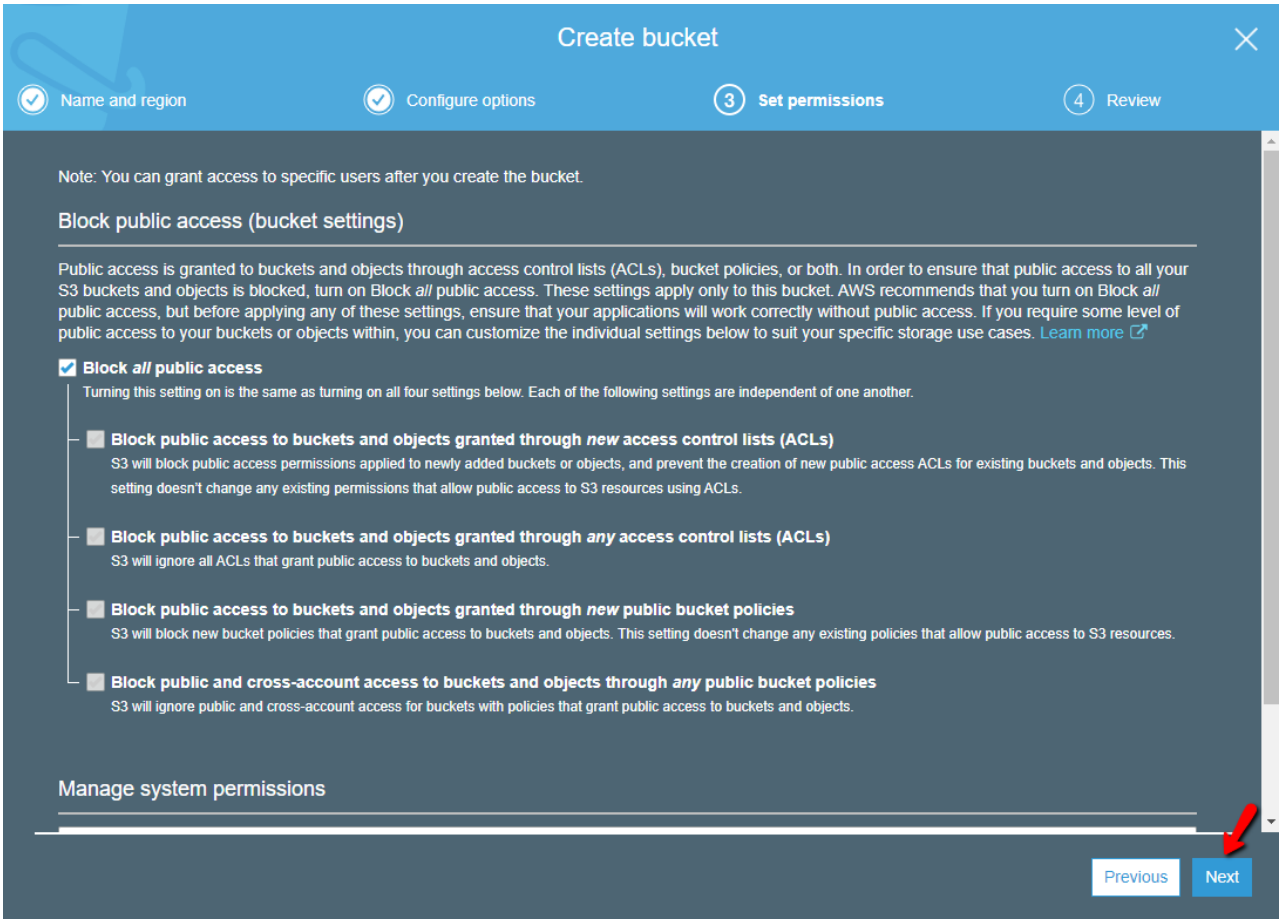
1. Go to the https://console.aws.amazon.com/s3/.
2. Choose **Create bucket**.



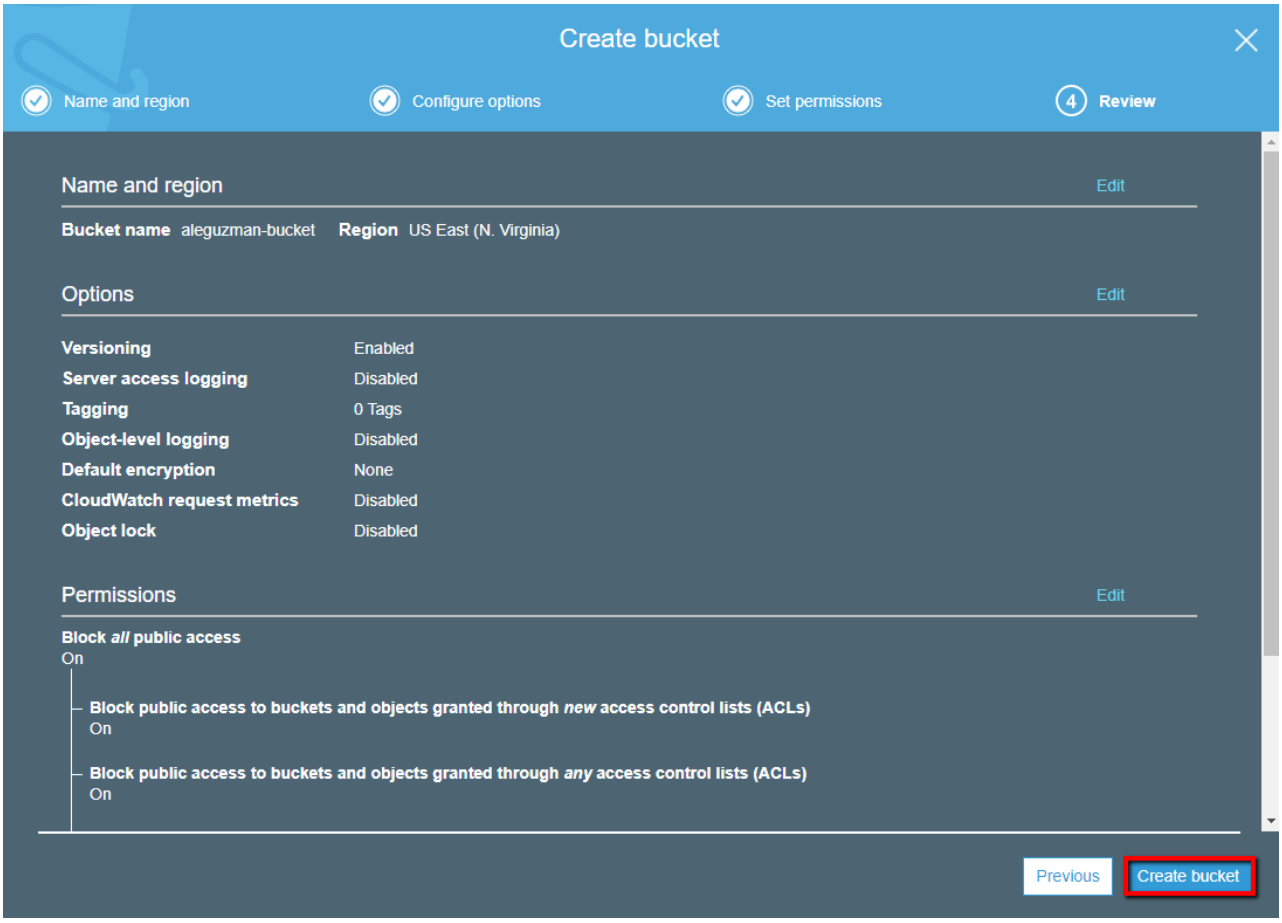
3. Type a bucket name, and then choose **Next**.
4. Select **Versioning** to keep all versions in the same bucket, and then choose **Next**.



5. Choose **Next** to accept the default permissions.



6. Choose **Create bucket**.

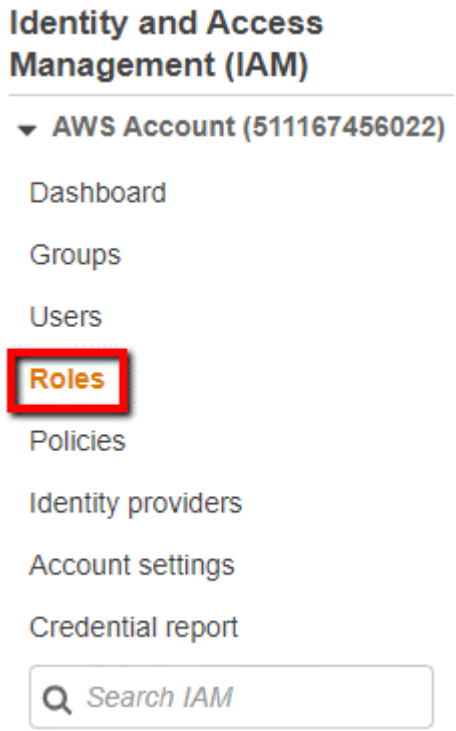


2.2 Create an OTA Update Service Role

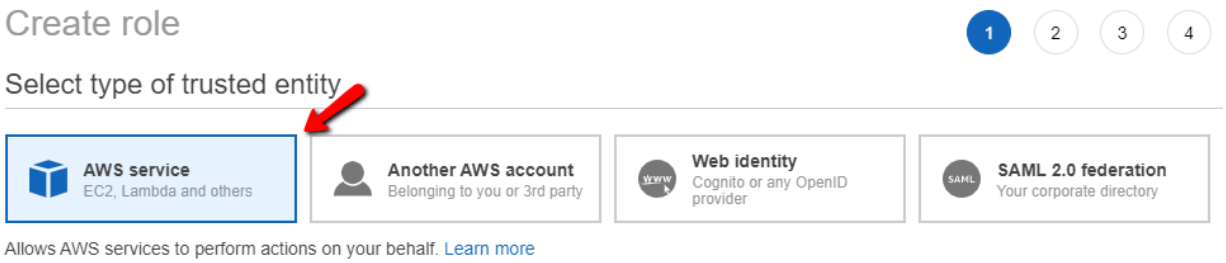
2.2.1 Create an OTA service role

1. Sign in to the https://console.aws.amazon.com/iam/.

2. From the navigation pane, choose **Roles**.



3. Choose to **Create role**.
4. Under **Select type of trusted entity**, choose **AWS Service**.



5. Choose **IoT** from the list of AWS services.

Allows AWS services to perform actions on your behalf. Learn more

Choose the service that will use this role

EC2
Allows EC2 instances to call AWS services on your behalf.

Lambda
Allows Lambda functions to call AWS services on your behalf.

API Gateway	CodeDeploy	ElastiCache	Lambda	S3
AWS Backup	Comprehend	Elastic Beanstalk	Lex	SMS
AWS Chatbot	Config	Elastic Container Service	License Manager	SNS
AWS Support	Connect	Elastic Transcoder	Machine Learning	SWF
Amplify	DMS	ElasticLoadBalancing	Macie	SageMaker
AppStream 2.0	Data Lifecycle Manager	Forecast	MediaConvert	Security Hub
AppSync	Data Pipeline	Global Accelerator	Migration Hub	Service Catalog
Application Auto Scaling	DataSync	Glue	OpsWorks	Step Functions
Application Discovery Service	DeepLens	Greengrass	Personalize	Storage Gateway
Batch	Directory Service	GuardDuty	QLDB	Textract
CloudFormation	DynamoDB	Inspector	RAM	Transfer
CloudHSM	EC2	IoT	RDS	Trusted Advisor
CloudTrail	EC2 - Fleet	IoT Things Graph	Redshift	VPC
CloudWatch Application Insights	EC2 Auto Scaling	KMS	Rekognition	WorkLink
CloudWatch Events	EKS	Kinesis	RoboMaker	WorkMail
CodeBuild	EMR			

6. Under **Select your use case**, choose **IoT**.

Select your use case

IoT
Allows IoT to call AWS services on your behalf.

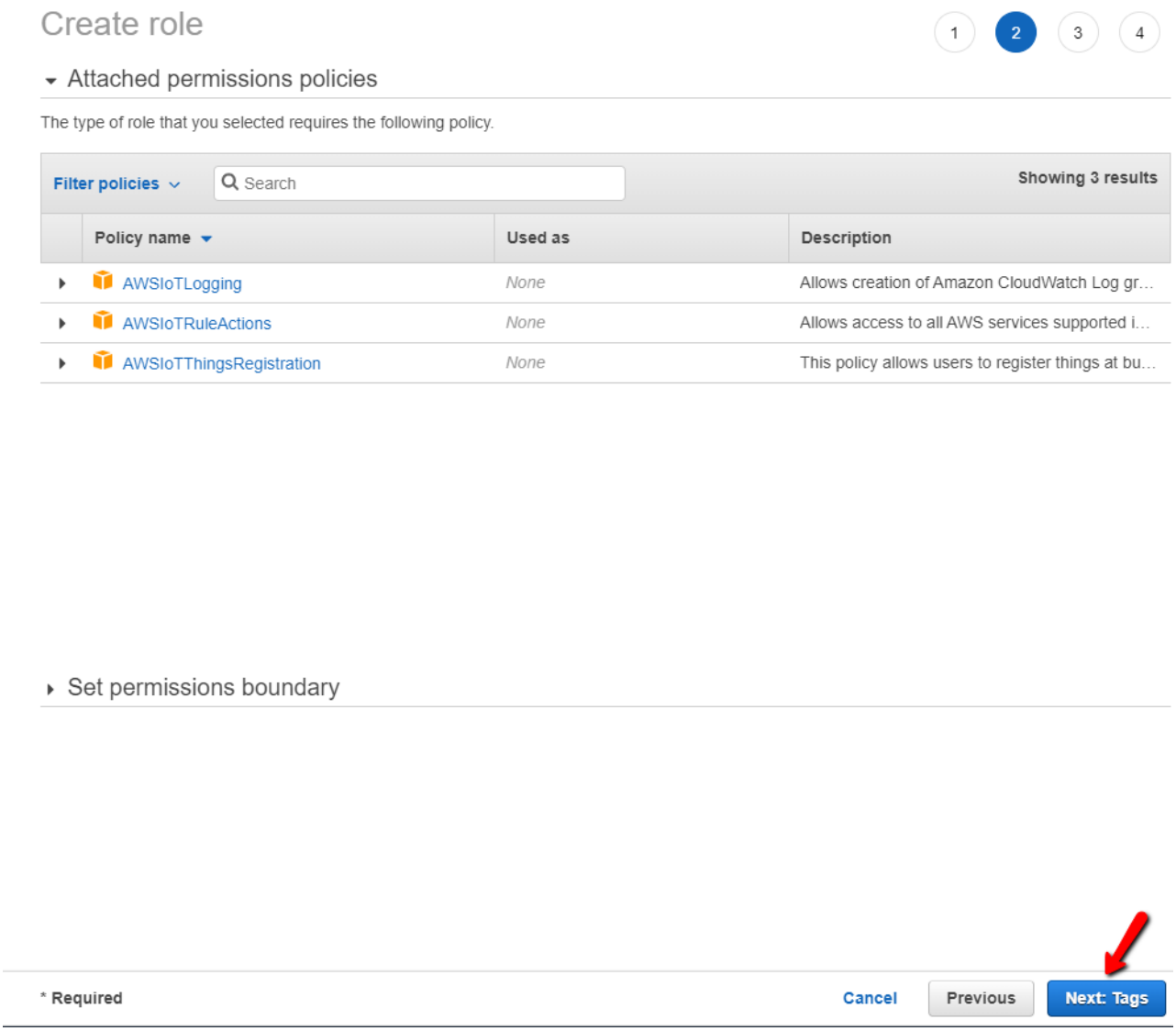
IoT - Device Defender Audit
Provides AWS IoT Device Defender read access to IoT and related resources.

IoT - Device Defender Mitigation Actions
Provides AWS IoT Device Defender write access to IoT and related resources for execution of Mitigation Actions.

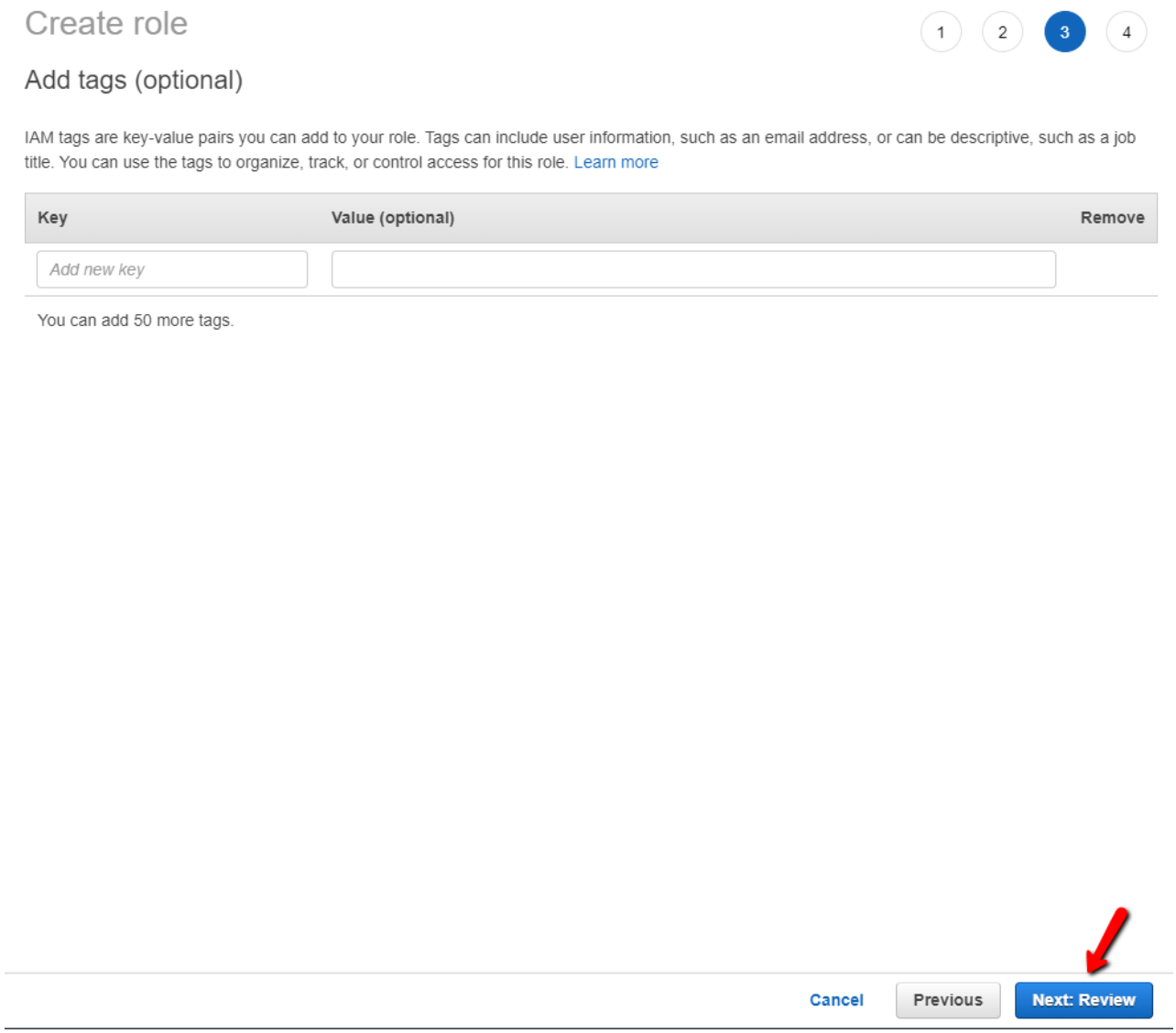
7. Choose **Next: Permissions**.

Next: Permissions

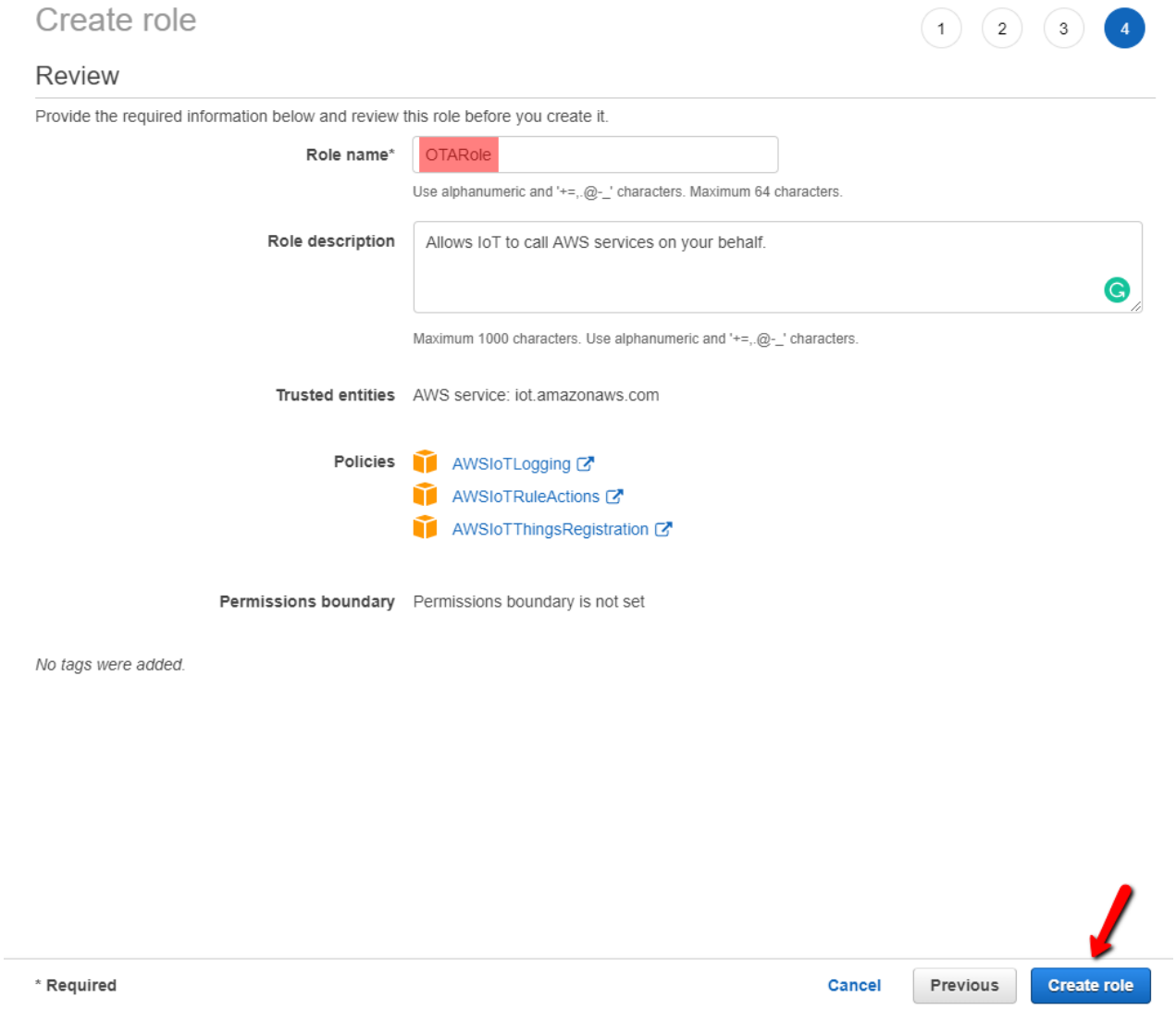
8. Choose **Next: Tags**.



9. Choose **Next: Review**.

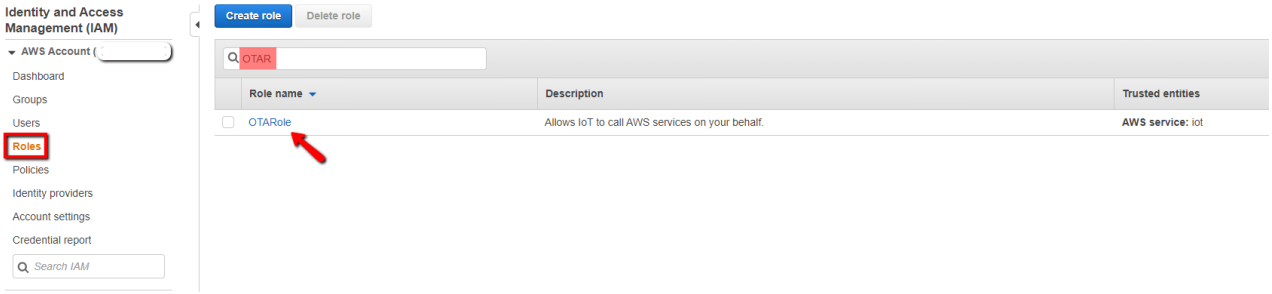


10. Enter a role name and description and then choose to **Create role**.

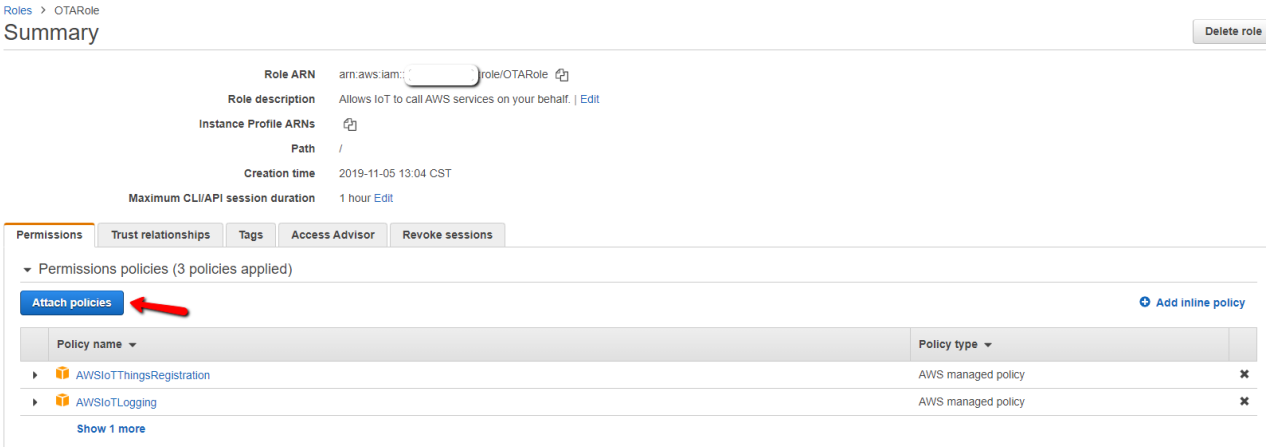


2.2.2 To add OTA update permissions to your OTA service role

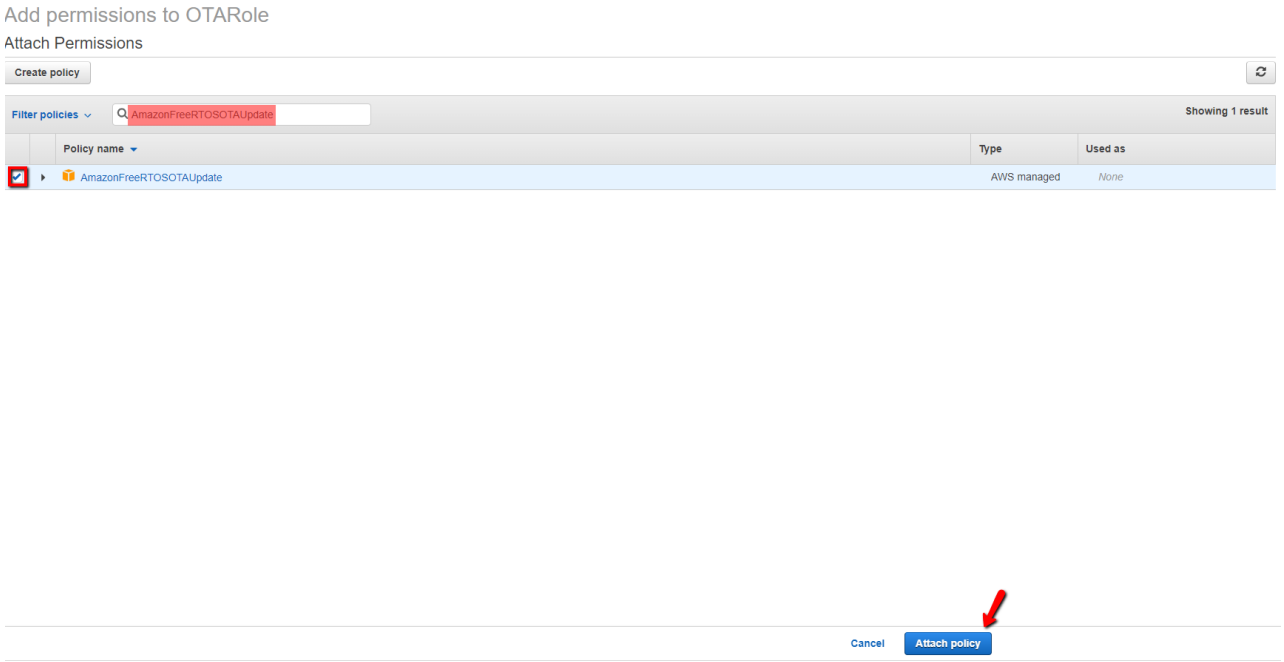
1. In the search box on the IAM console page, enter the name of your role, and then choose it from the list.



2. Choose **Attach policies**.

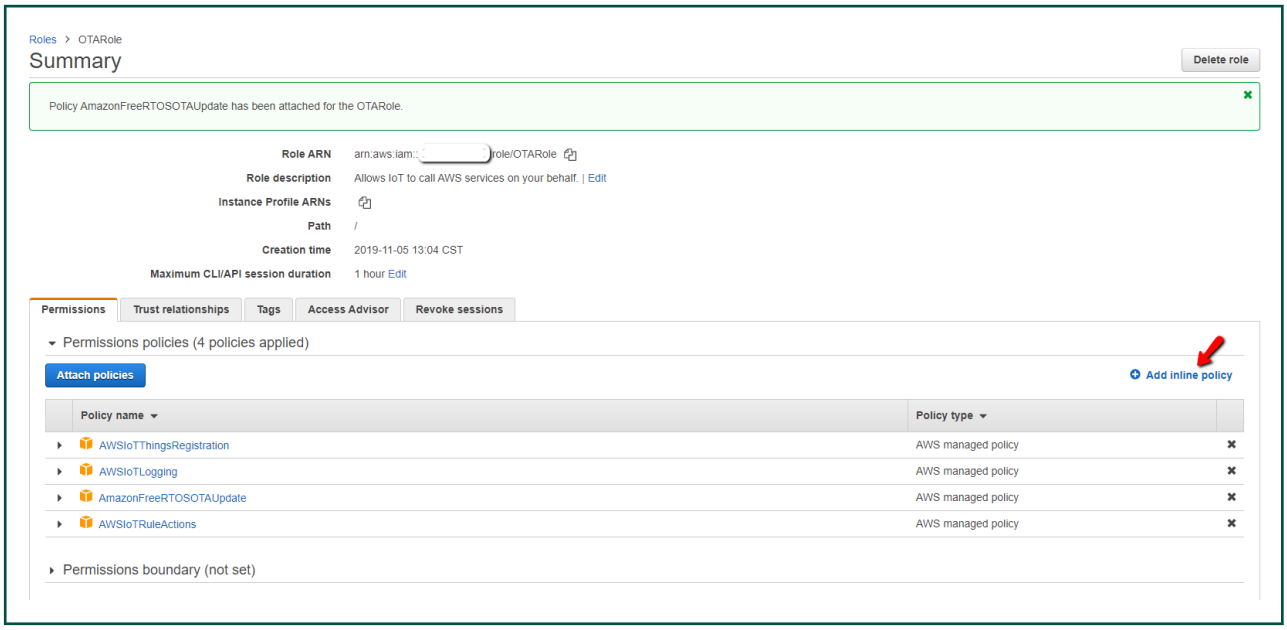


3. In the **Search** box, enter **AmazonFreeRTOSOTAUpdate**, select **AmazonFreeRTOSOTAUpdate** from the list of filtered policies, and then choose **Attach policy** to attach the policy to your service role.



2.2.3 To add the required IAM permissions to your OTA service role

1. Choose **Add inline policy**.



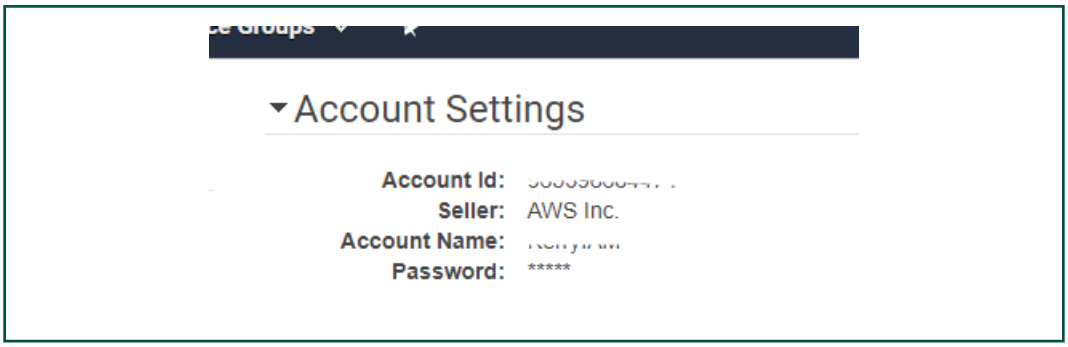
2. Choose the **JSON** tab.
3. Copy and paste the following policy document into the text box:

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "iam:GetRole",
            "iam:PassRole"
         ],
         "Resource": "arn:aws:iam::<your_account_id>:role/<your_role_name>"
      }
    ]
}
```

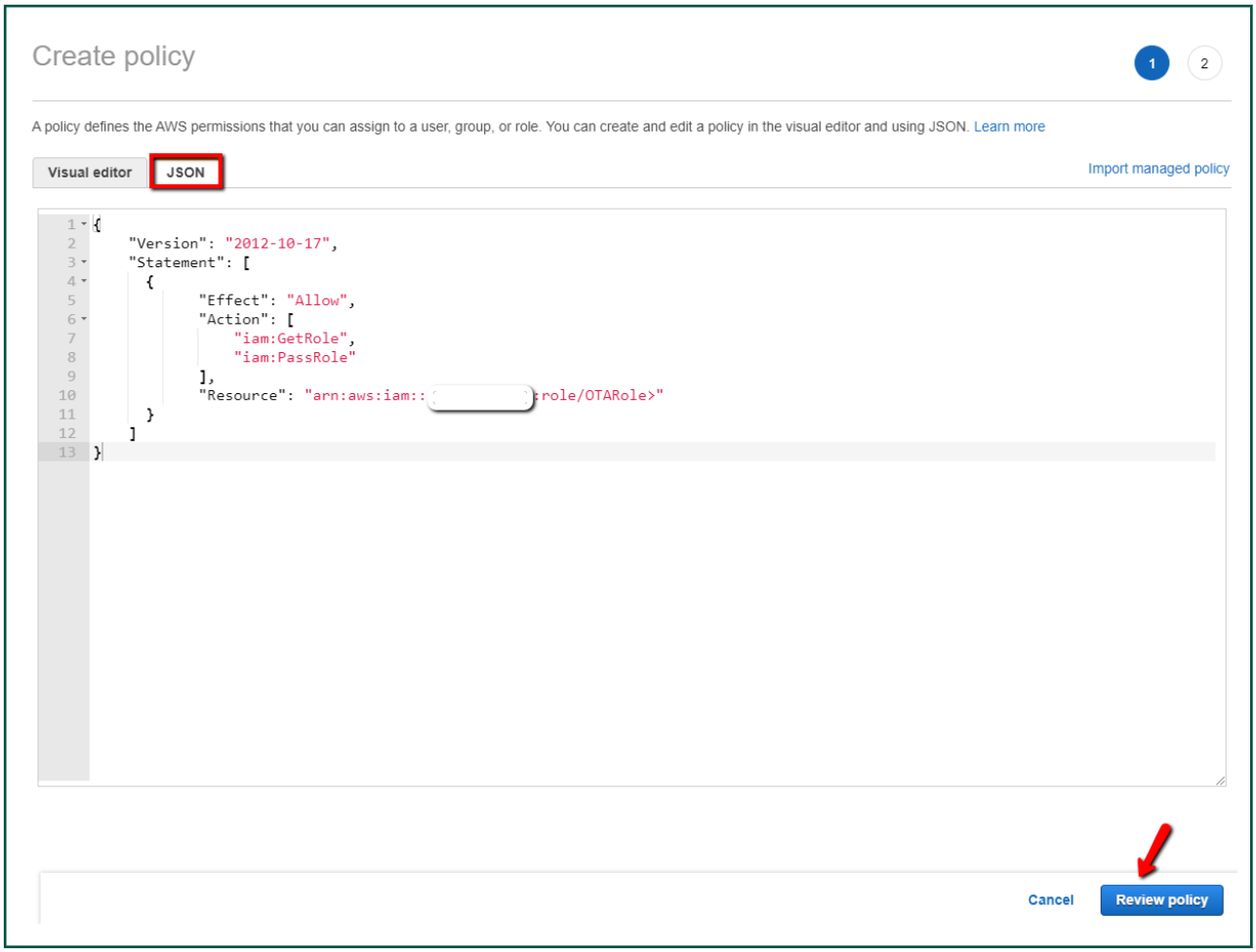
Make sure that you replace *<your_account_id>* with your AWS account ID, and *<your_role_name>* with the name of the OTA service role.

NOTE

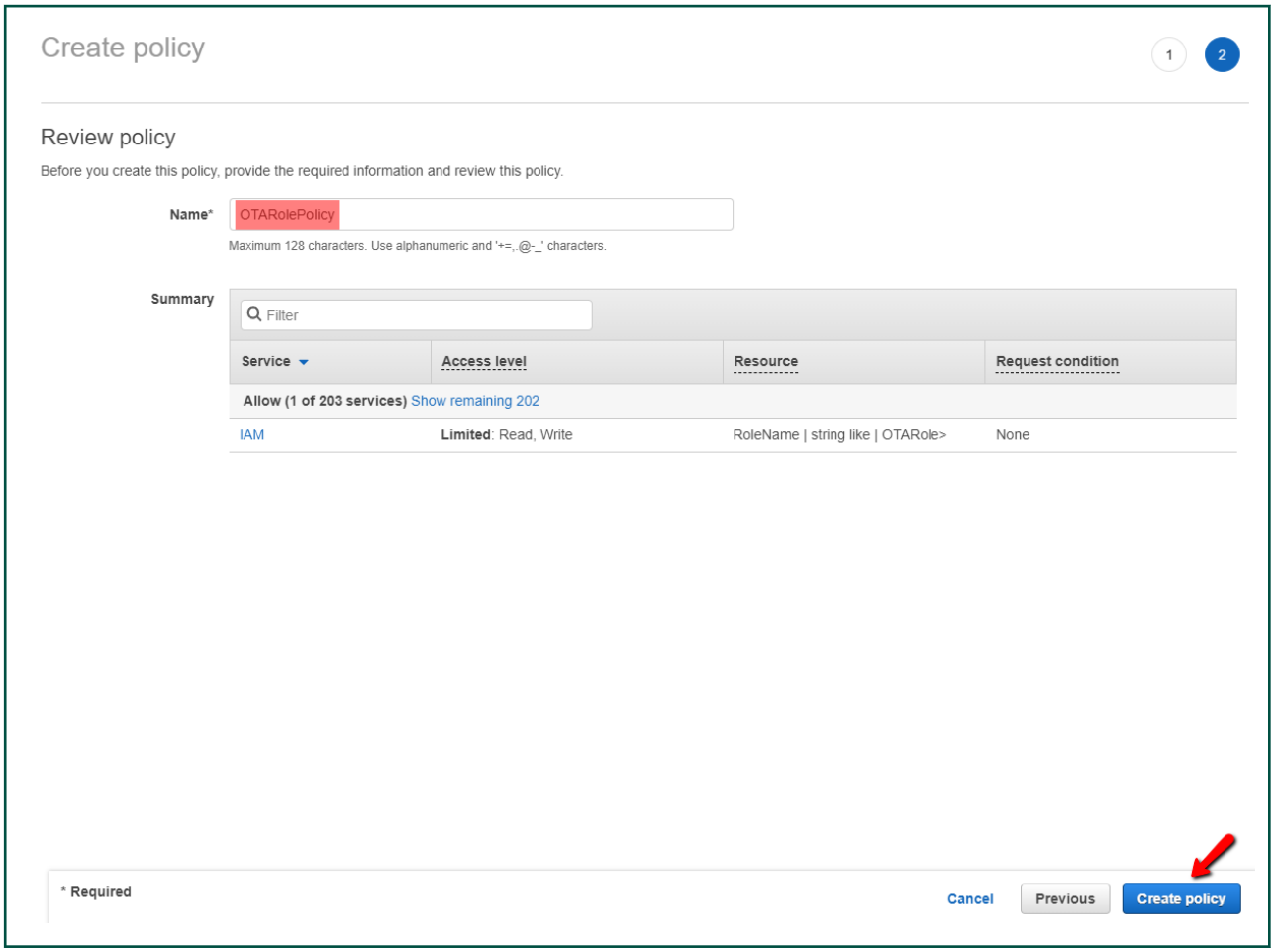
To obtain account ID, select account name in Web page menu bar and select **My account** from the drop-down menu. Make note of the **Account ID** under **Account Settings**.



4. Choose **Review policy**.

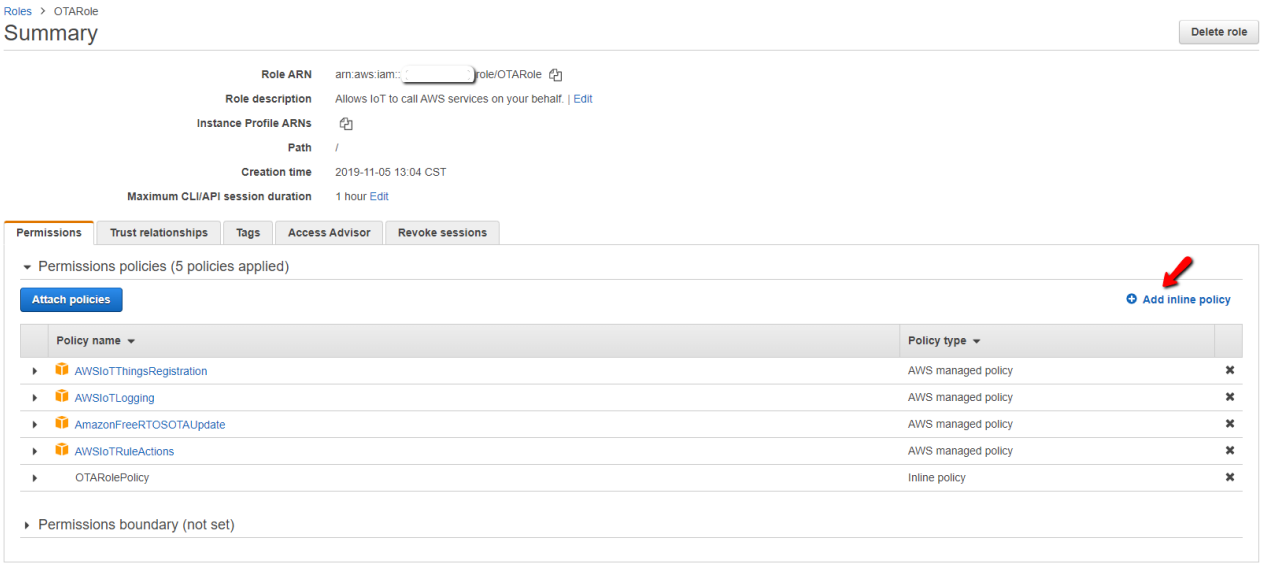


5. Enter a name for the policy, and then choose **Create policy**.



2.2.4 To add the required Amazon S3 permissions to your OTA service role

1. In the search box on the IAM console page, enter the name of your role, and then choose it from the list.
2. Choose **Add inline policy**.



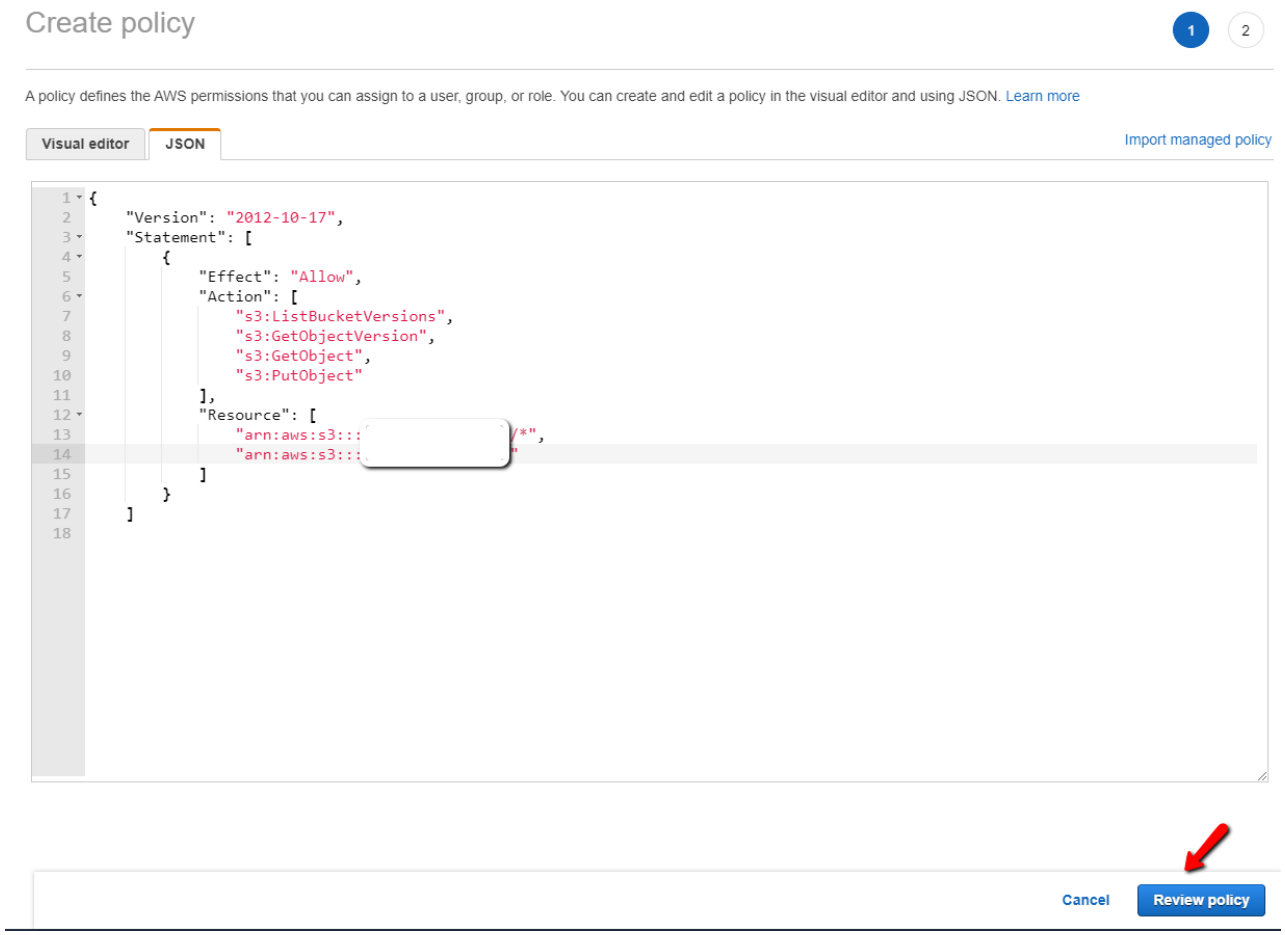
3. Choose the **JSON** tab.

Copy and paste the following policy document into the box:

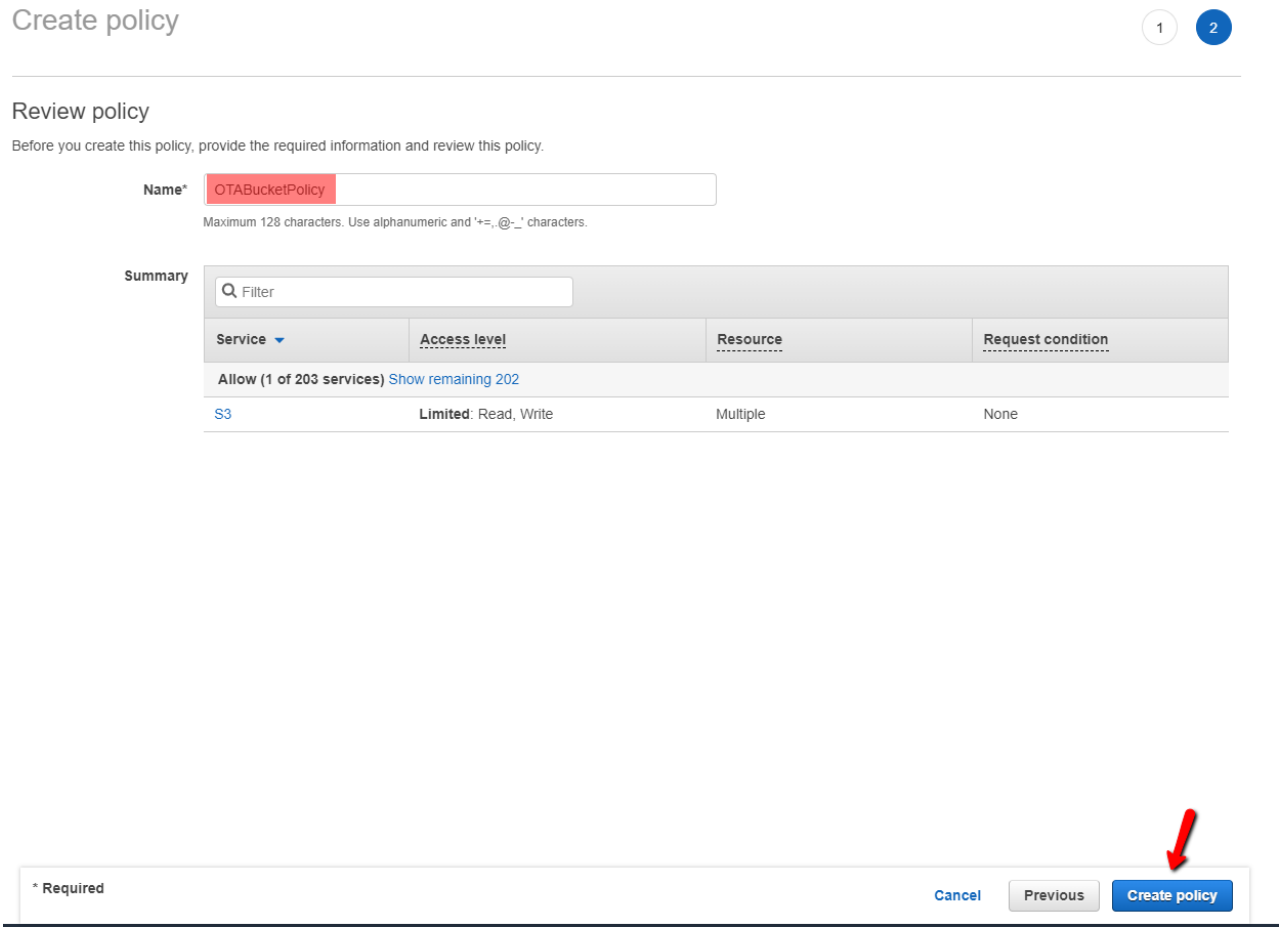
```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "s3:ListBucketVersions",
            "s3:GetObjectVersion",
         "s3:GetObject",
         "s3:PutObject"
         ],
         "Resource": [
            "arn:aws:s3:::<example-bucket>/*"
            "arn:aws:s3:::<example-bucket>"
        ]
      }
    ]
}
```

This policy grants your OTA service role permission to read Amazon S3 objects. Make sure that you replace *<example-bucket>* with the name of your bucket.

1. Choose **Review policy**.

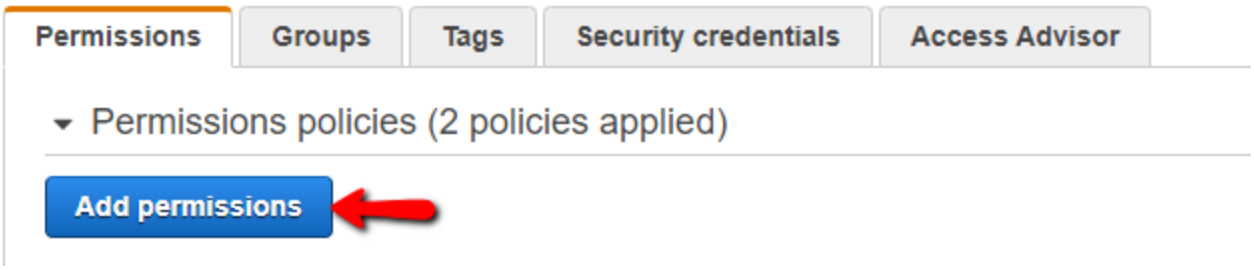


2. Enter a name for the policy, and then choose **Create policy**.

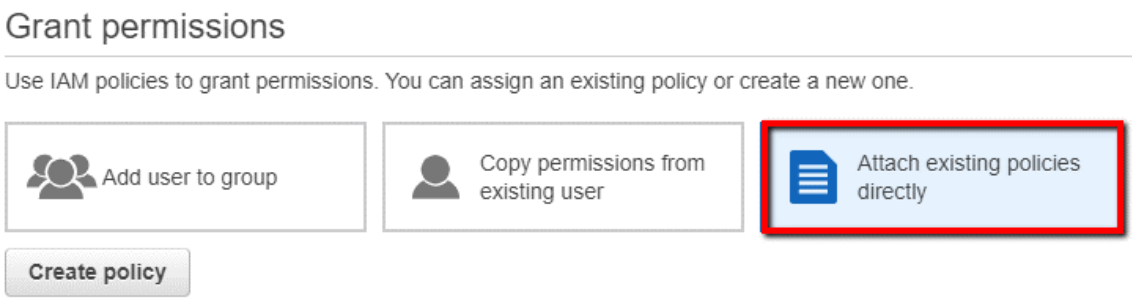


2.3 Create an OTA User Policy

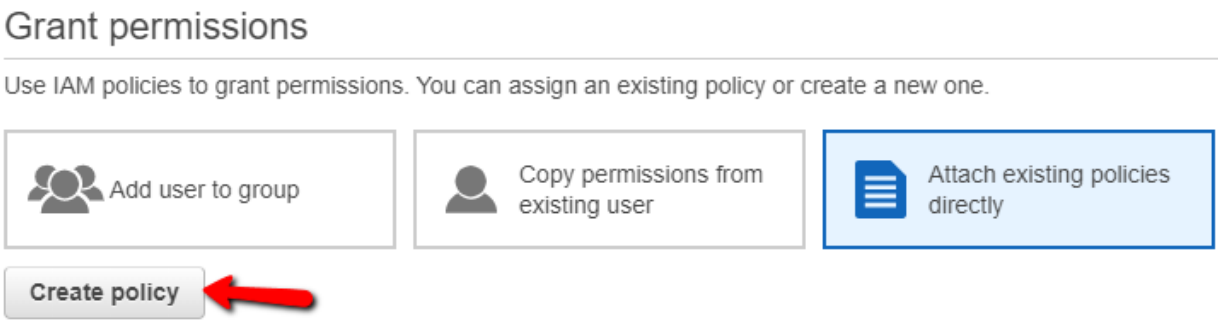
1. Open the https://console.aws.amazon.com/iam/ console.
2. In the navigation pane, choose **Users**.
3. Choose your IAM user from the list.
4. Choose **Add permissions**.



5. Choose **Attach existing policies directly**.



6. Choose **Create policy**.



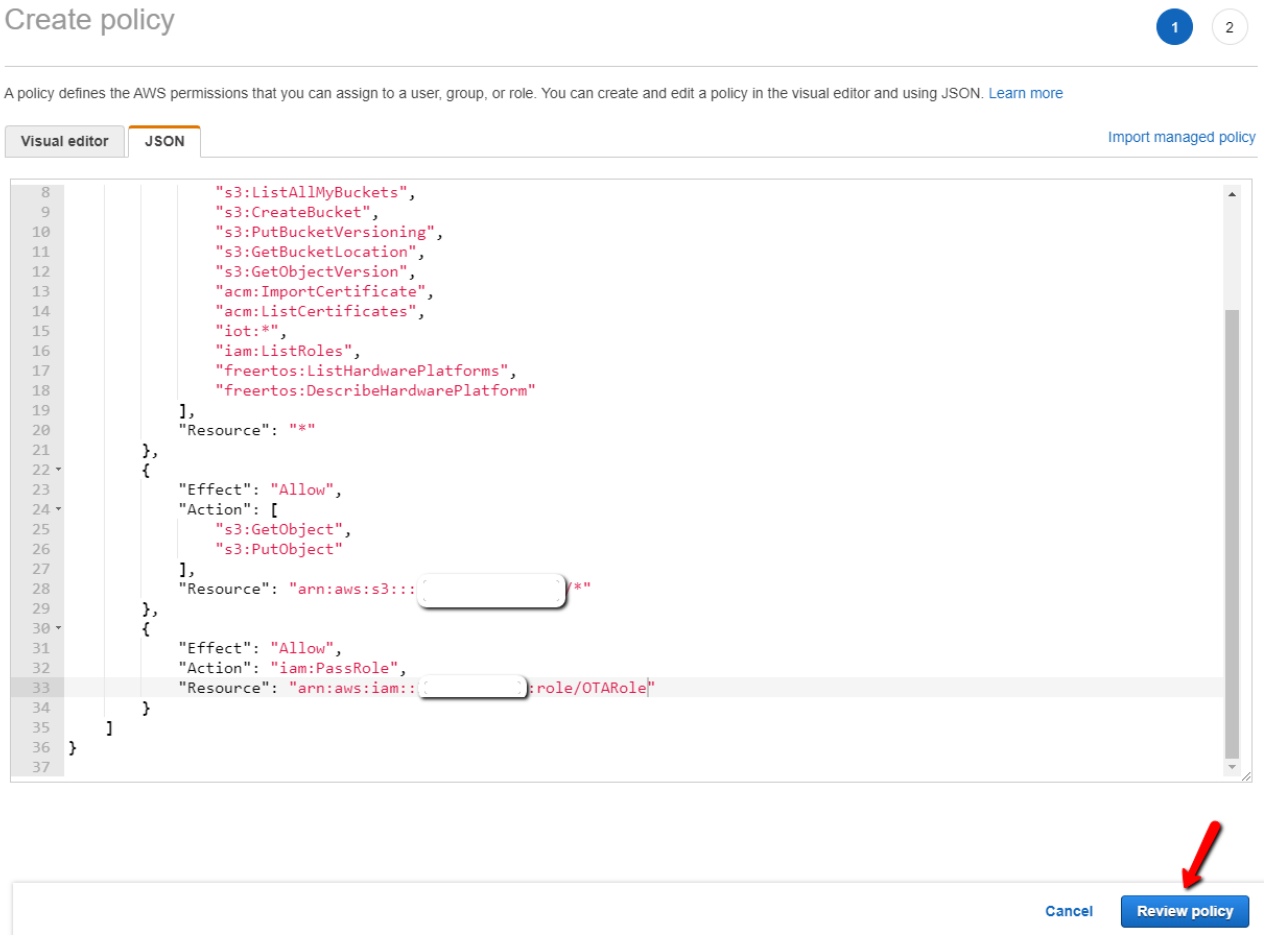
Choose the **JSON** tab, and copy and paste the following policy document into the policy editor:

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "s3:ListBucket",
            "s3:ListAllMyBuckets",
            "s3:CreateBucket",
            "s3:PutBucketVersioning",
            "s3:GetBucketLocation",
            "s3:GetObjectVersion",
            "acm:ImportCertificate",
            "acm:ListCertificates",
            "iot:*",
            "iam:ListRoles",
            "freertos:ListHardwarePlatforms",
            "freertos:DescribeHardwarePlatform"
         ],
         "Resource": "*"
      }
      {
         "Effect": "Allow",
         "Action": [
            "s3:GetObject",
            "s3:PutObject"
         ],
         "Resource": "arn:aws:s3:::<example-bucket>/*"
      }
      {
         "Effect": "Allow",
```

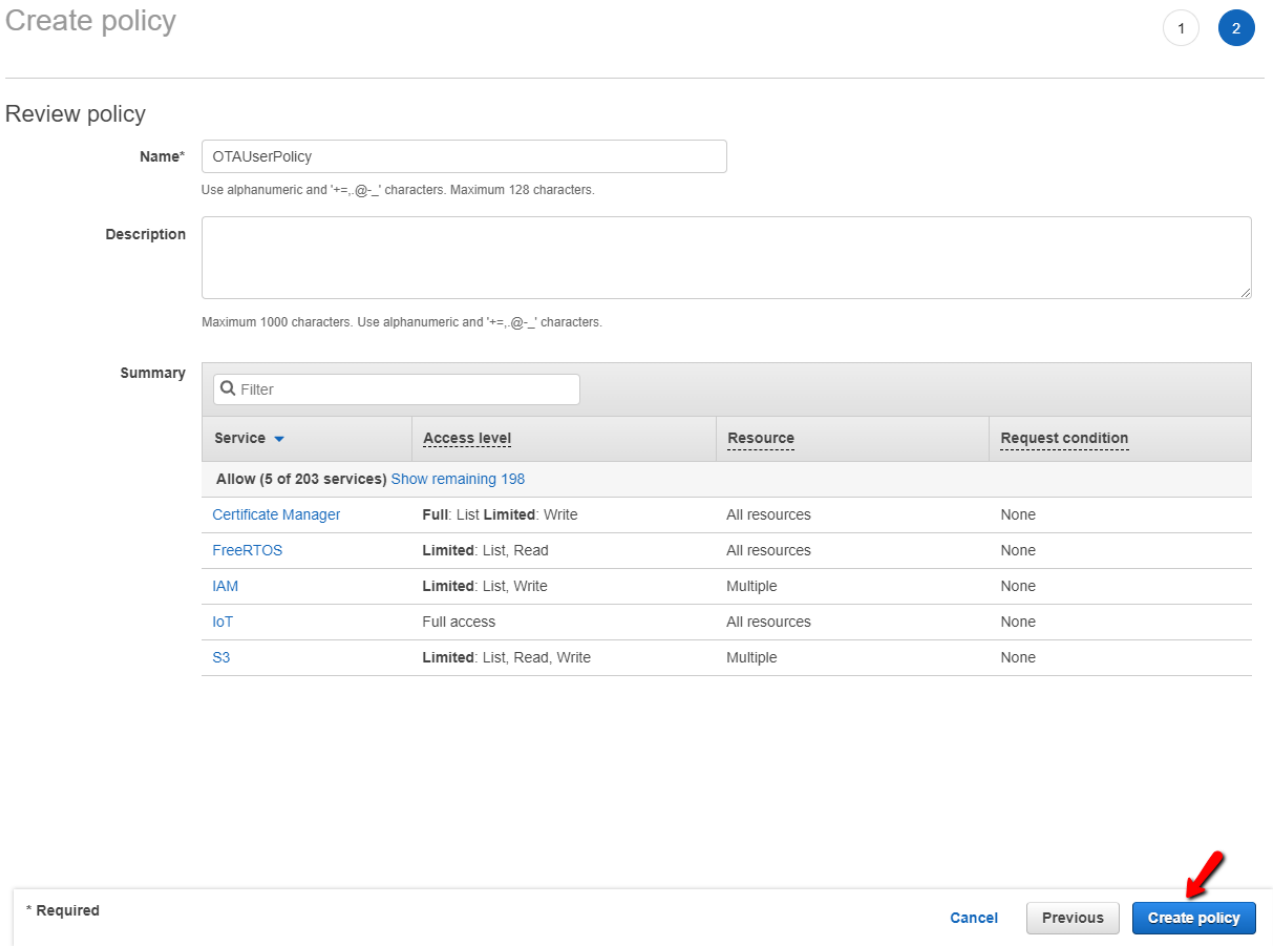
```
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::<your-account-id>:role/<role-name>"
        }
    ]
}
```

Replace *<example-bucket>* with the name of the Amazon S3 bucket where your OTA update firmware image is stored. Replace *<your-account-id>* with your AWS account ID. You can find your AWS account ID in the upper right of the console. When you enter your account ID, remove any dashes (-). Replace *<role-name>* with the name of the IAM service role you just created.

1. Choose **Review policy**.



2. Enter a name for your new OTA user policy, and then choose **Create policy**.

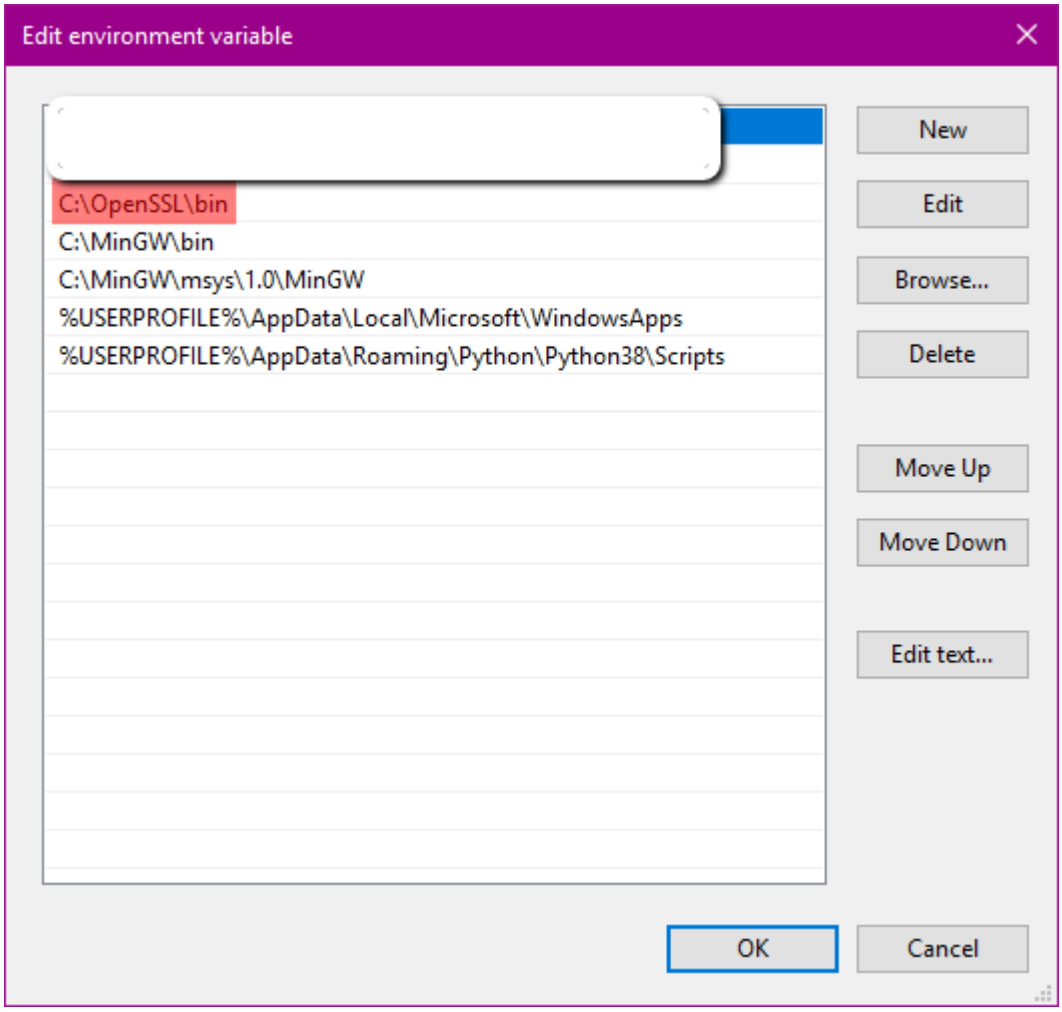


2.4 Windows Pre-Requisites

2.4.1 OpenSSL

1. Install OpenSSL

2. Modify the system environment variable path to add your OpenSSL bin directory

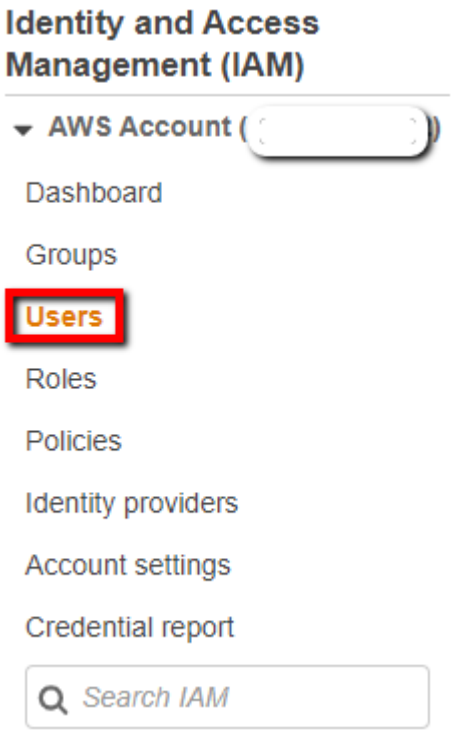


Make sure openssl gets assigned to the OpenSSL executable in your command prompt or terminal environment

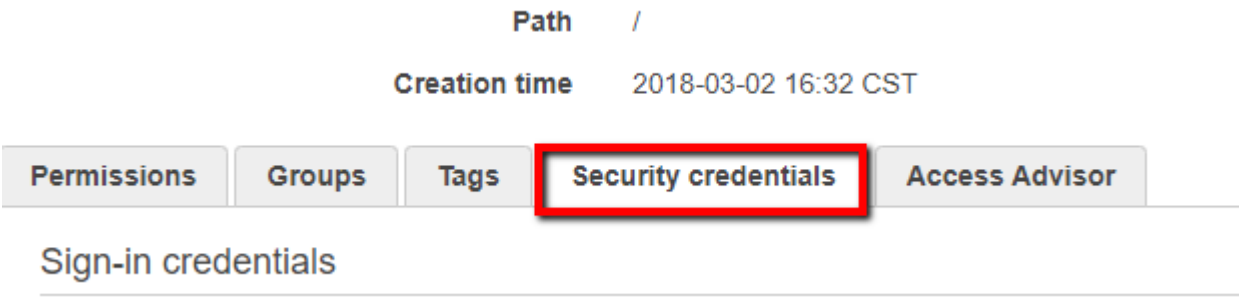
2.4.2 Install the AWS CLI

1. Follow the instructions for AWS CLI bundler installer https://docs.aws.amazon.com/cli/latest/userguide/install-windows.html#install-msi-on-windows
2. Go to the IAM console https://console.aws.amazon.com/iam/

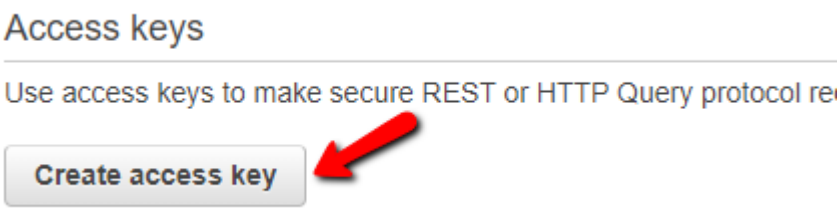
3. In the navigation pane, choose **Users**.



4. Choose your IAM user account.
5. Select **Security credentials**

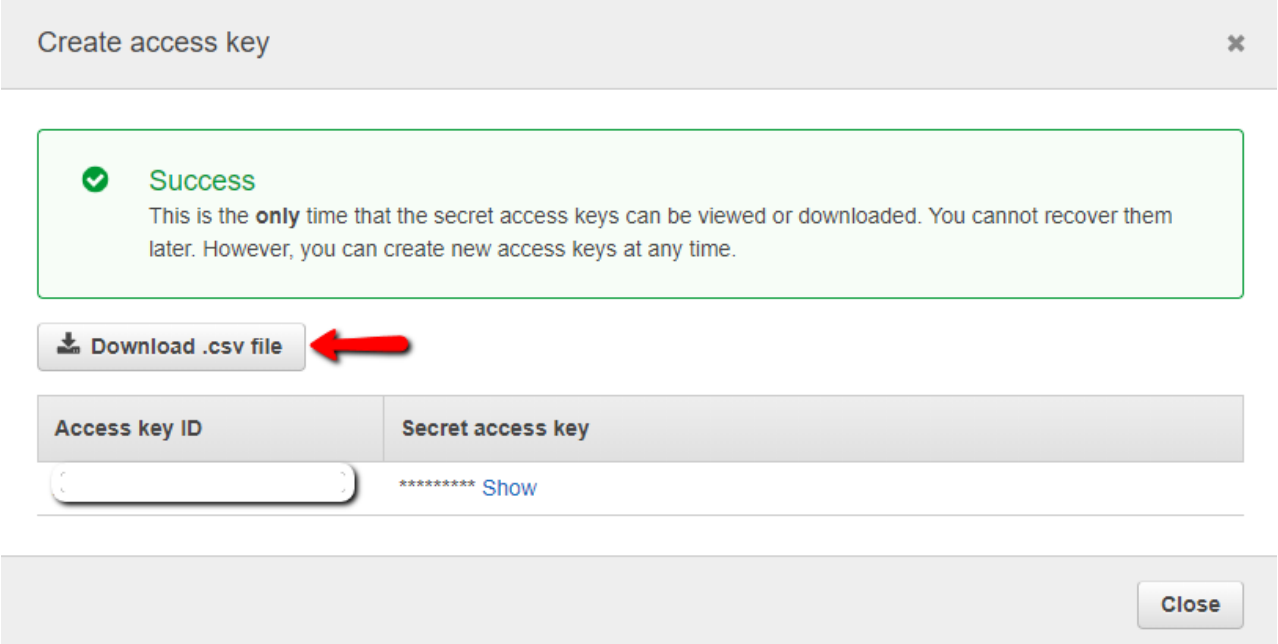


6. In the **Acess keys** section, choose **Create access key**.

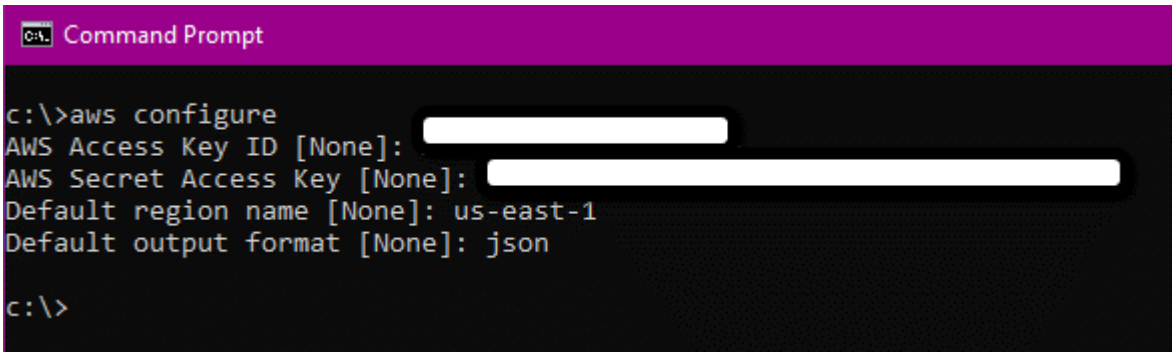


7. To view the new access key pair, choose **Show**. You will not have access to the secret access key again after this dialog box closes. Your credentials will look something like this: *Access key ID: AKIAIOSFODNN7EXAMPLE Secret access key: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY*
8. To download the key pair, choose **Download .csv** file. Store the keys in a secure location. You will not have access to the secret access key again after this dialog box closes. Keep the keys confidential in order to protect your AWS account and

never email them. Do not share them outside your organization, even if an inquiry appears to come from AWS or Amazon.com. No one who legitimately represents Amazon will ever ask you for your secret key.



9. After you download the .csv file, choose **Close**. When you create an access key, the key pair is active by default, and you can use the pair right away.
10. For general use, the aws configure command is the fastest way to set up your AWS CLI installation



2.5 Creating a Code-Signing Certificate

1. In your working directory, use the following text to create a file named cert_config.txt.
   Replace *test_signer@amazon.com* with your email address: [req]

prompt = no

distinguished_name = my_dn

[my_dn]

commonName = test_signer@amazon.com

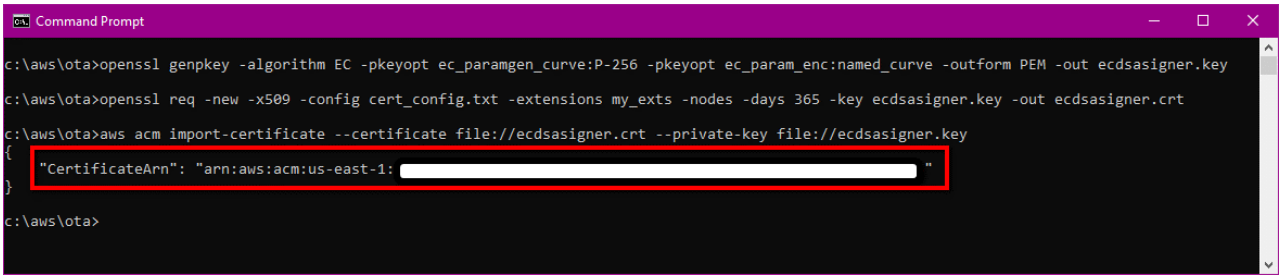
[my_exts]

keyUsage = digitalSignature

extendedKeyUsage = codesigning

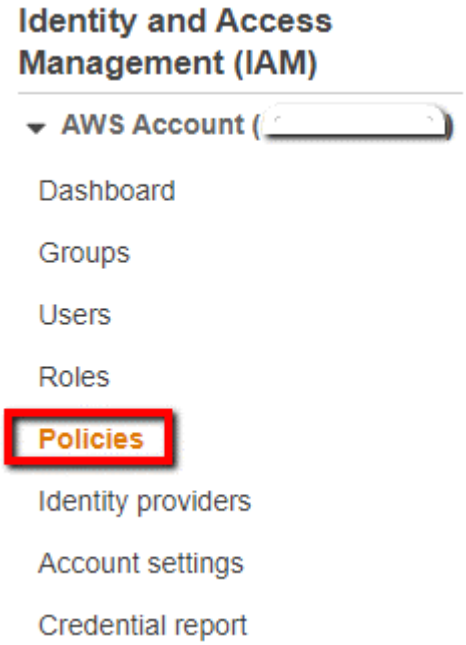
1. Using openSSL command line create an ECDSA code-signing private key: openssl genpkey -algorithm EC -pkeyopt ec_paramgen_curve:P-256 -pkeyopt ec_param_enc:named_curve -outform PEM -out ecdsasigner.key

2. Create an ECDSA code-signing certificate: openssl req -new -x509 -config cert_config.txt -extensions my_exts -nodes -days 365 -key ecdsasigner.key -out ecdsasigner.crt
3. Import the code-signing certificate, private key, and certificate chain into AWS Certificate Manager: aws acm import-certificate *--certificate file://ecdsasigner.crt --private-key file://ecdsasigner.key* ***Note:*** *this command displays an ARN for your certificate. Save it locally to use it while creating the OTA update job.*



Chapter 3
Grant access to code signing for AWS IoT

1. Sign in to the https://console.aws.amazon.com/iam/ .
2. In the navigation pane, choose **Policies**.

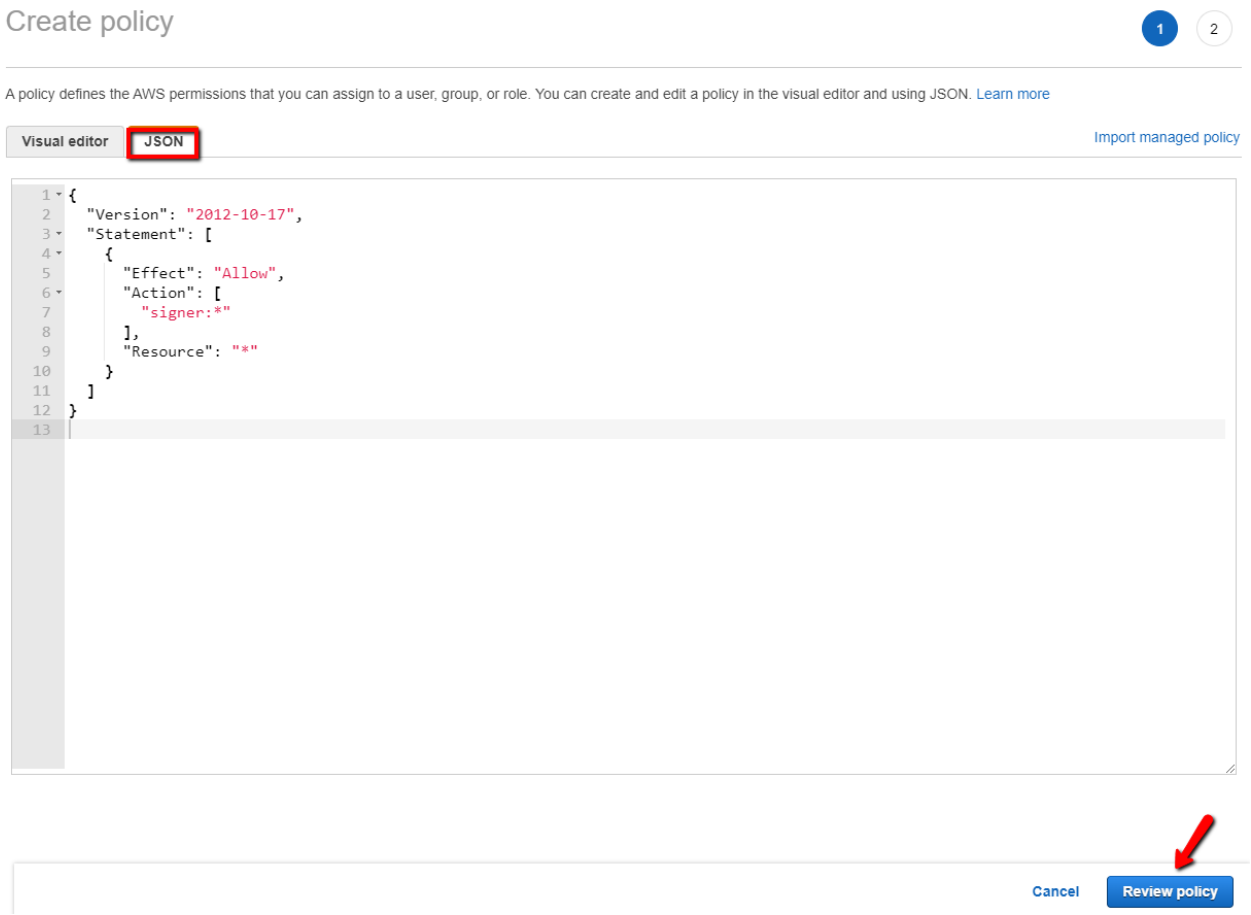


3. Choose **Create Policy**.

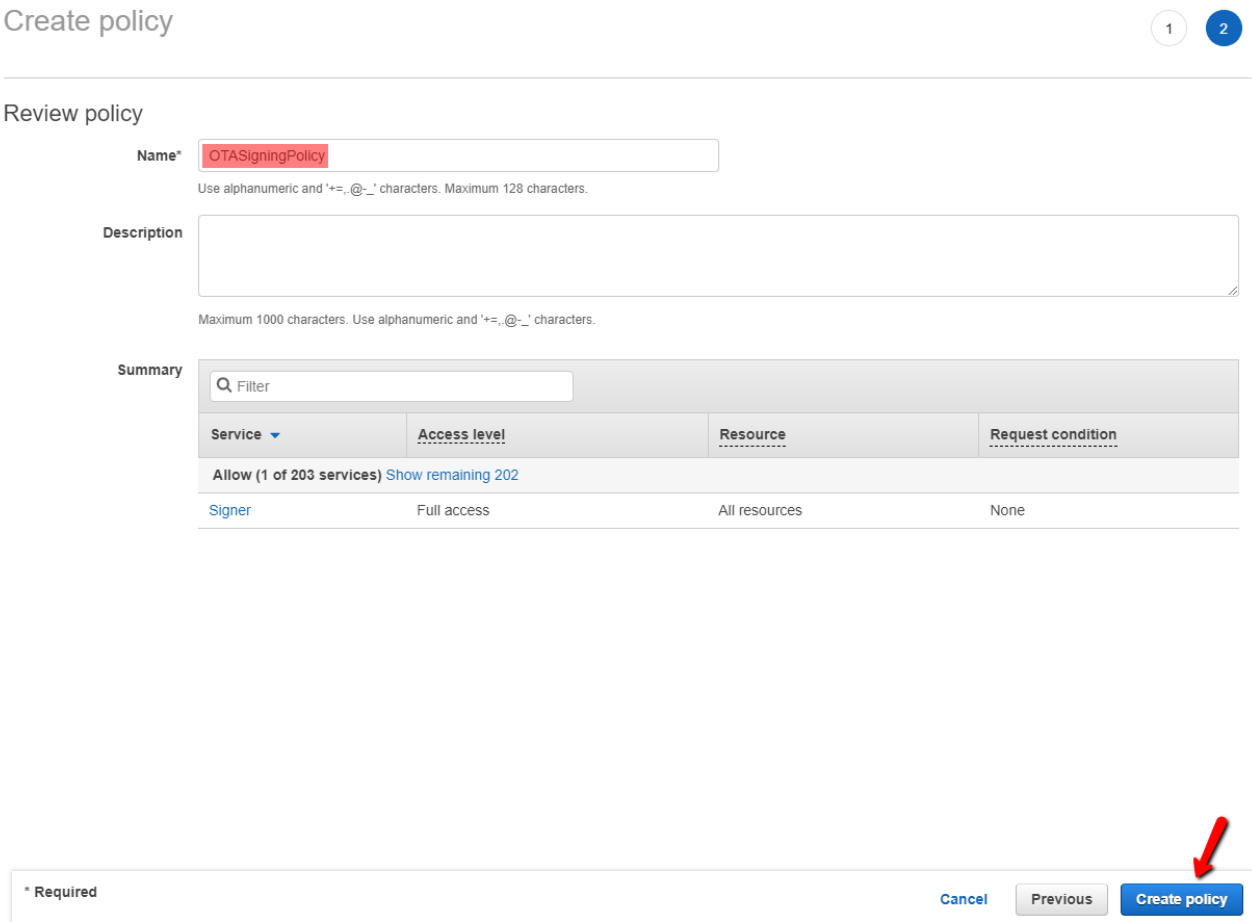
4. On the **JSON** tab, copy and paste the following JSON document into the policy editor. This policy allows the IAM user access to all code-signing operations.

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

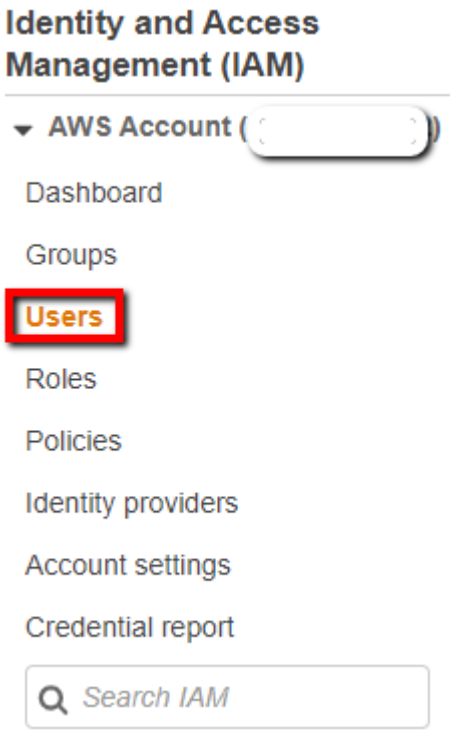
1. Choose **Review policy**.



2. Enter a policy name and description, and then choose **Create policy**.

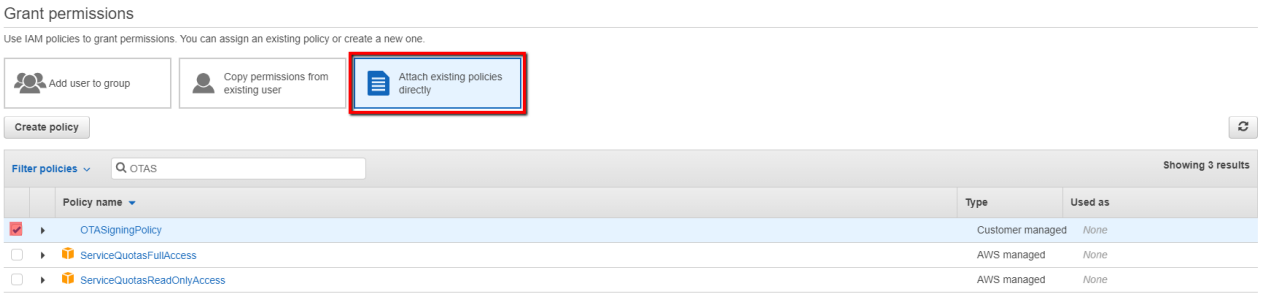


3. In the navigation pane, choose **Users**.



4. Choose your IAM user account.
5. On the **Permissions** tab, choose **Add permissions**.

6. Choose **Attach existing policies directly**, and then select the check box next to the code-signing policy you just created.



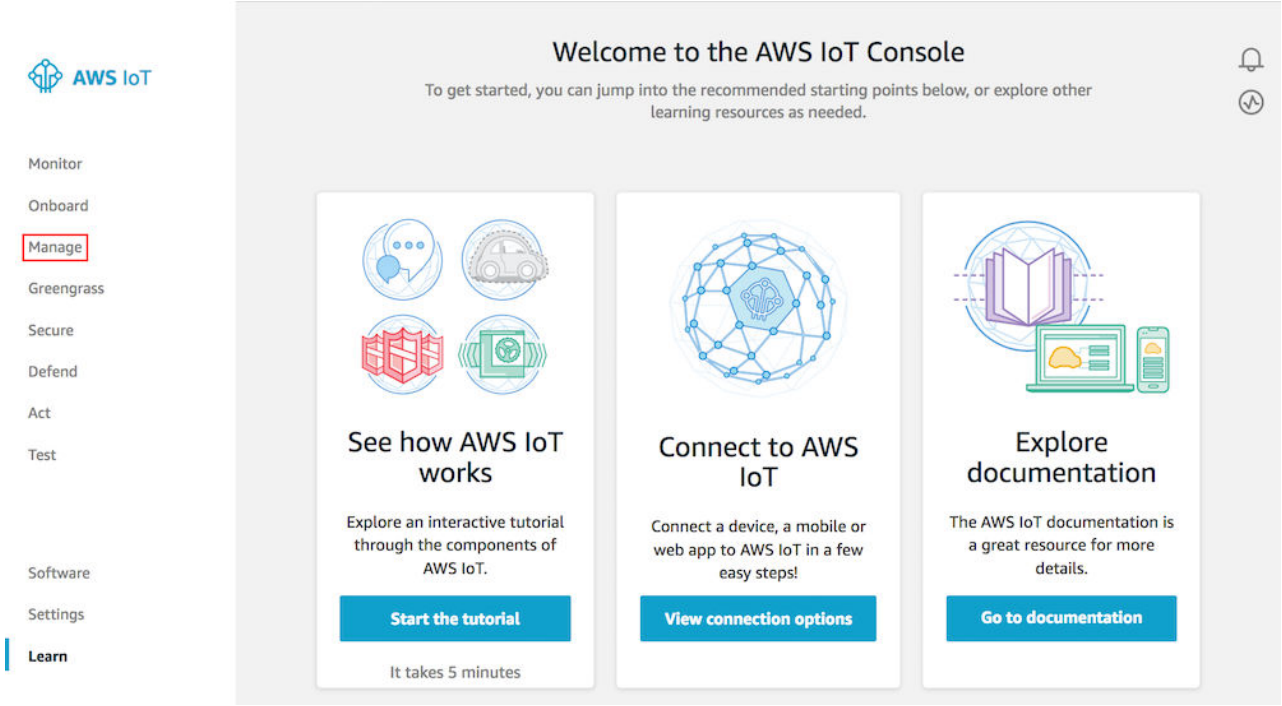
7. Choose **Next: Review**.

8. Choose **Add permissions**.

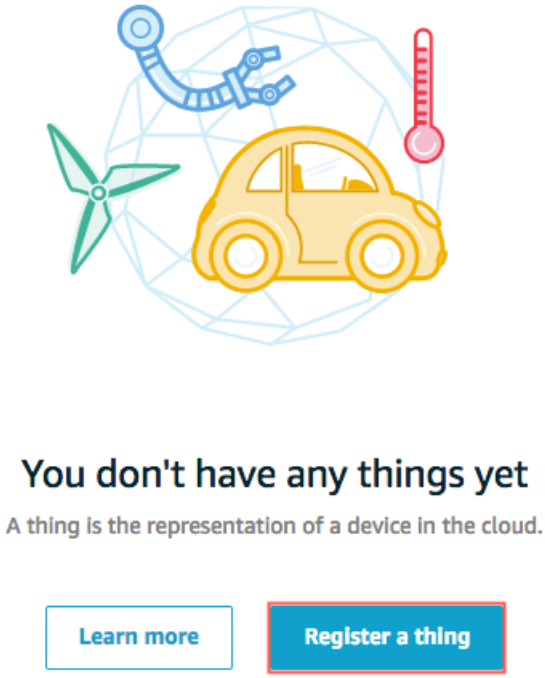
Chapter 4
AWS IoT

4.1 Create an AWS IoT Thing

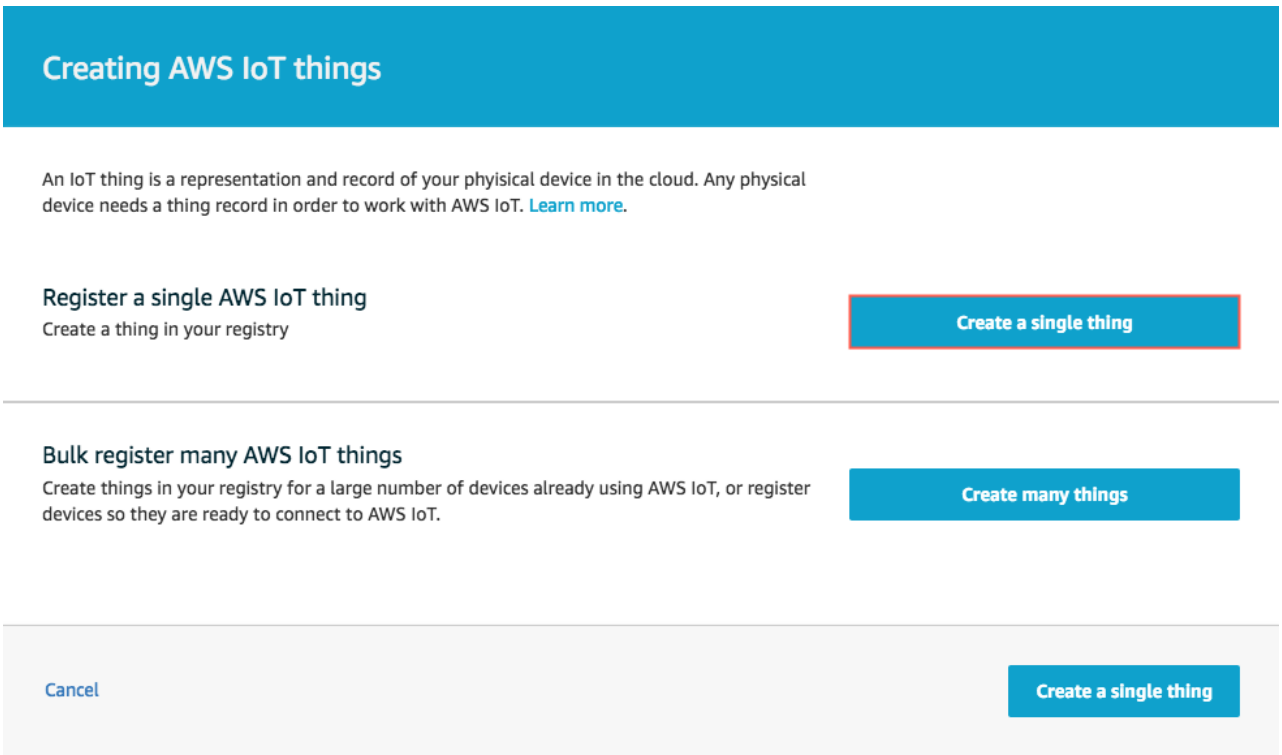
1. Open the AWS IoT console website https://console.aws.amazon.com/iot/
2. On the **Welcome to the AWS IoT Console** page, in the navigation pane, choose **Manage**.



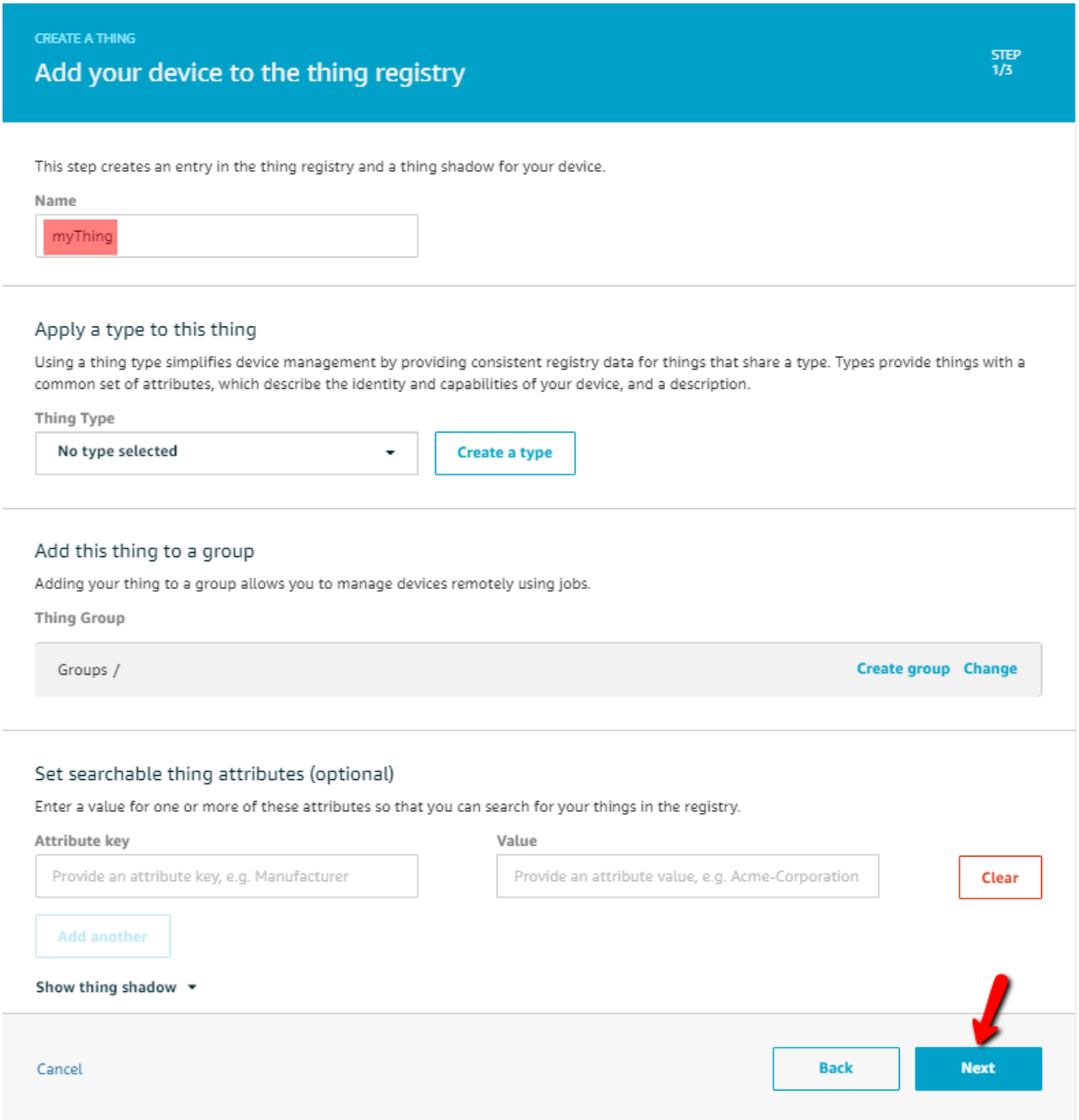
3. On the **You don't have any things yet** page, choose **Register a thing**.



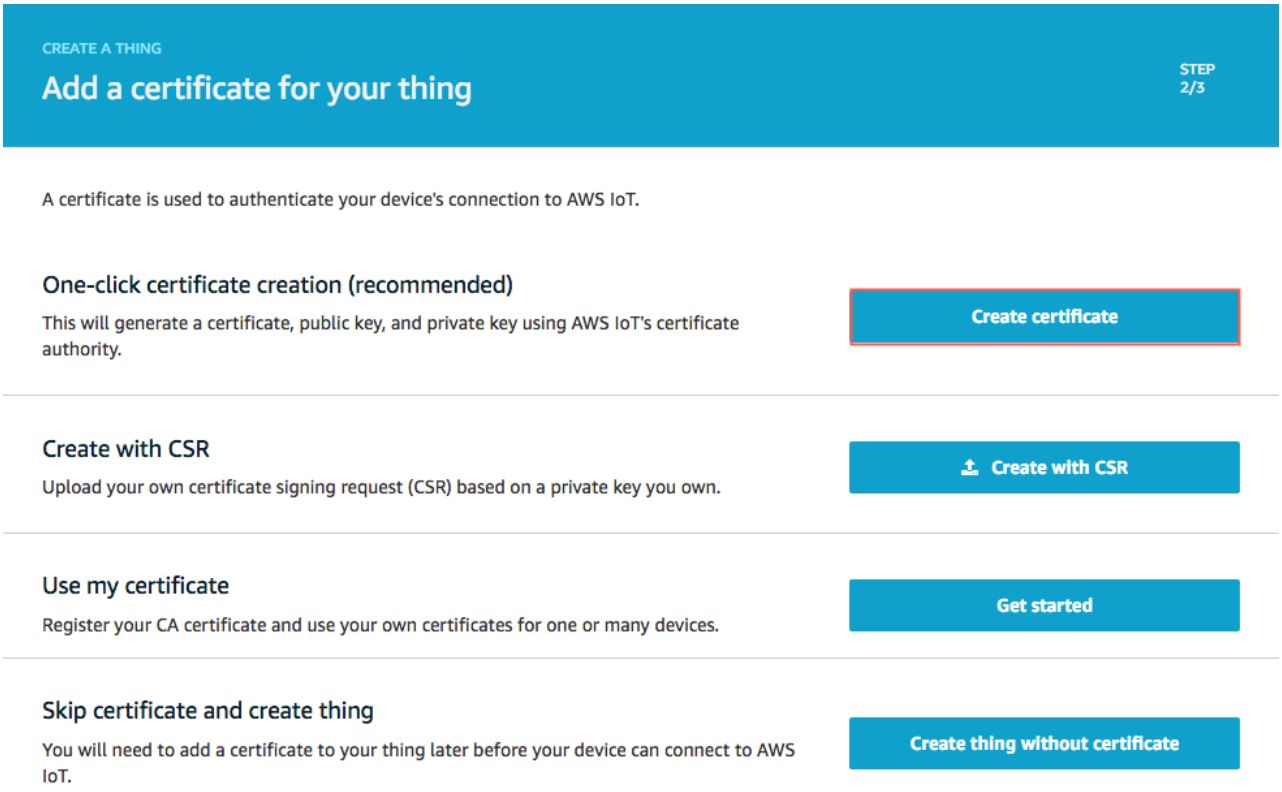
4. On the **Creating AWS IoT things** page, choose **Create a single thing**.



5. On the **Create a thing** page, in the **Name** field, enter a name for your thing, such as MyThing. Choose **Next**.

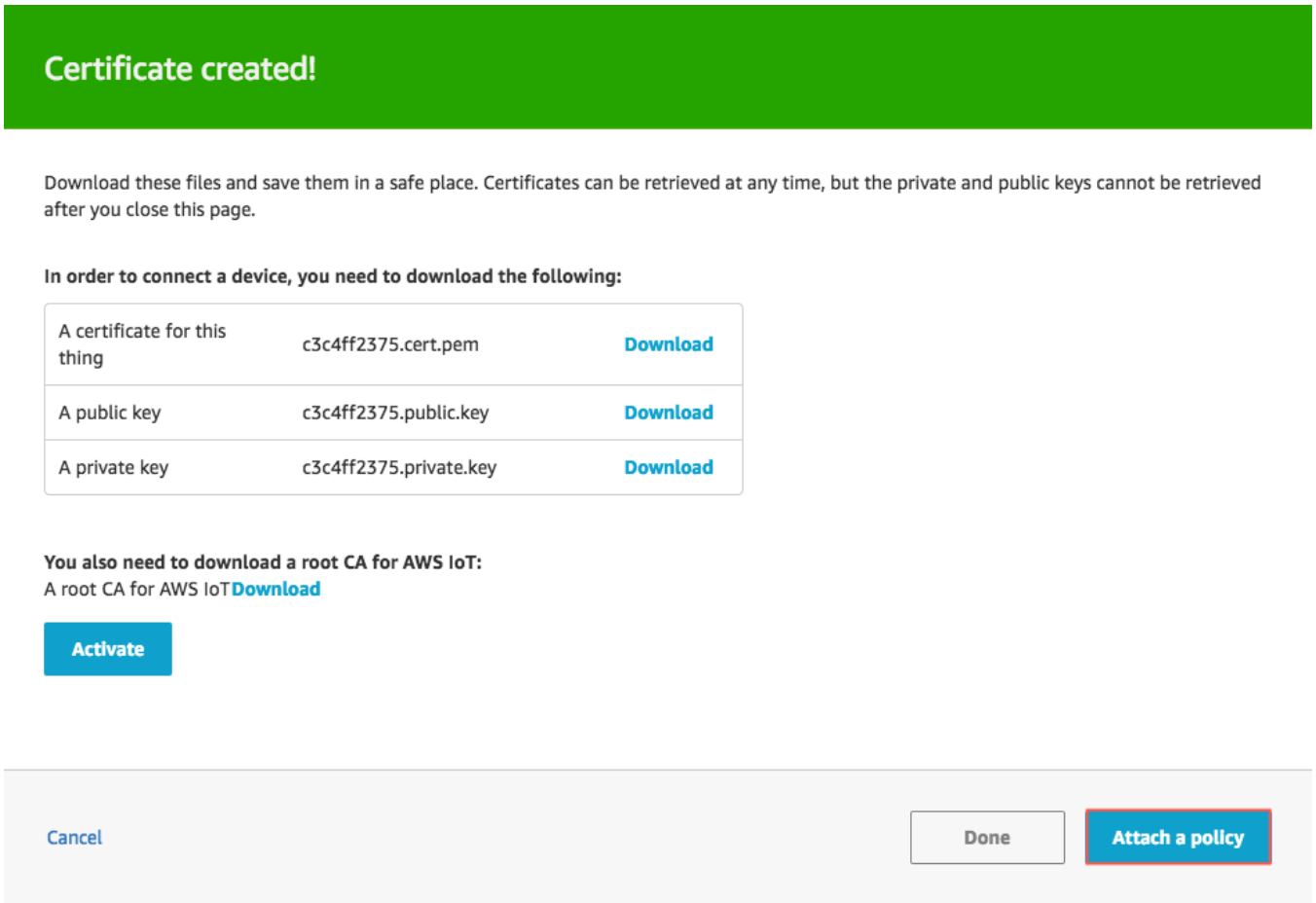


6. On the **Add a certificate for your thing** page, choose **Create certificate**. This generates an X.509 certificate and key pair.

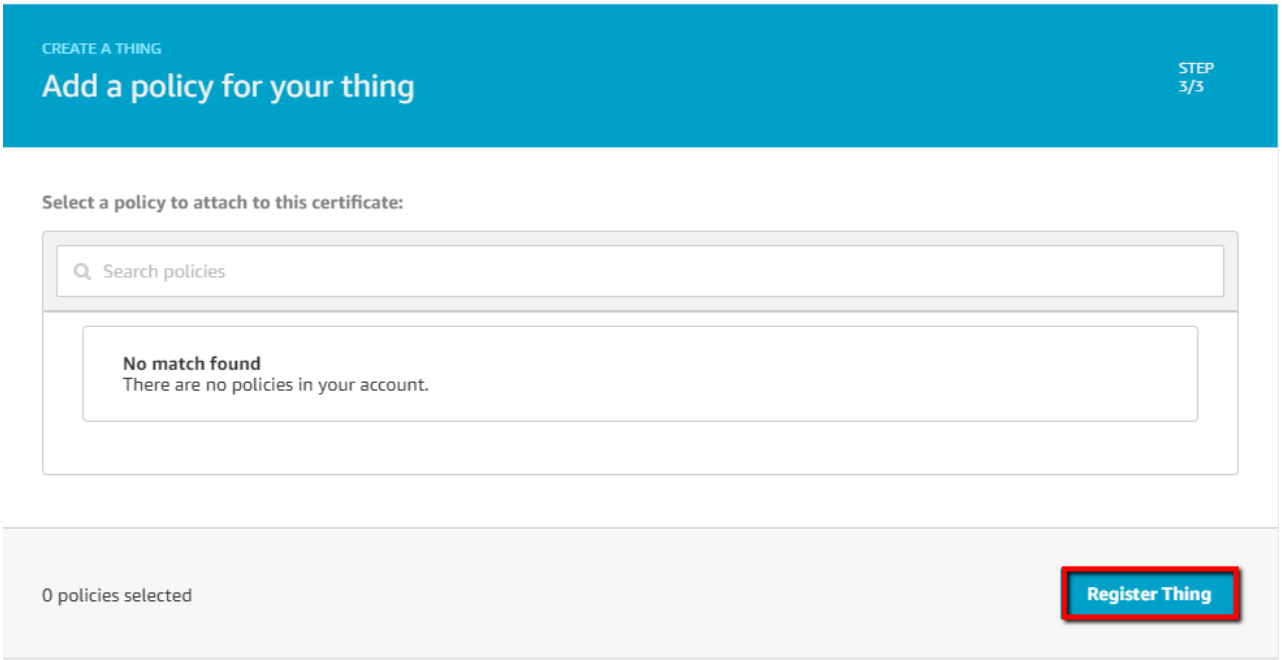


7. On the **Certificate created!** page, download your public and private keys, certificate, and root certificate authority (CA):
8. Choose **Download** for your certificate.
9. Choose **Download** for your private key.
10. Choose **Download** for the Amazon root CA. A new webpage is displayed. Choose **RSA 2048 bit key: Amazon Root CA 1**. This opens another webpage with the text of the root CA certificate. Copy this text and paste it into a file named `Amazon_Root_CA_1.pem`.

Most web browsers save downloaded files into a Downloads directory. You copy these files to a different directory when you run the sample applications. Choose **Activate** to activate the X.509 certificate, and then choose **Attach a policy**.

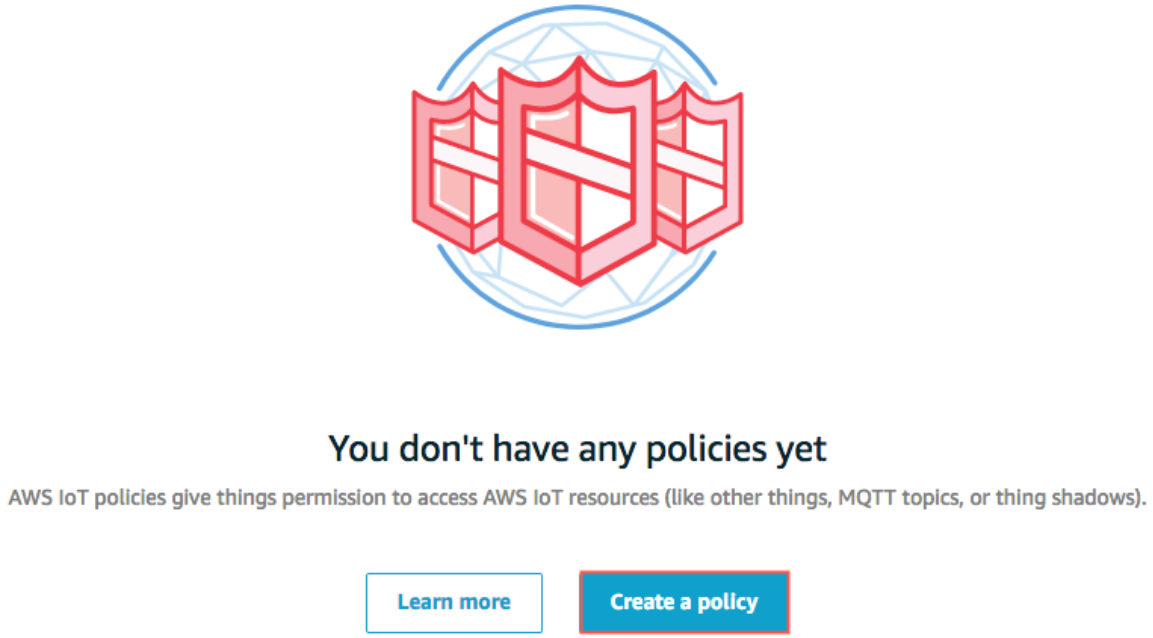


1. On the **Add a policy for your thing** page, choose **Register Thing**. After you register your thing, create and attach a new policy to the certificate.



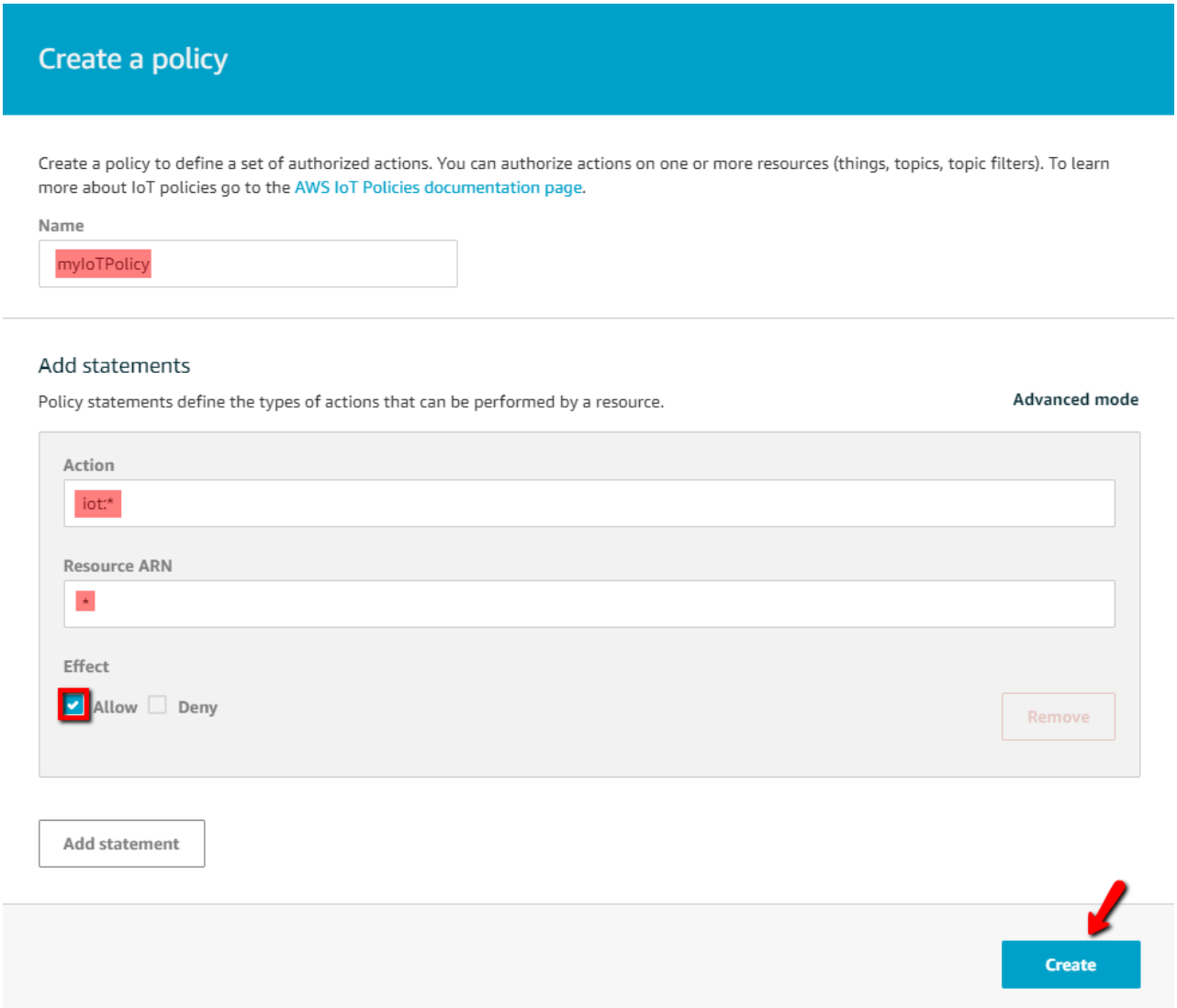
4.2 Create an AWS IoT Policy

1. In the left navigation pane, choose **Secure**, and then choose **Policies**. On the **You don't have a policy yet** page, choose **Create a policy.**



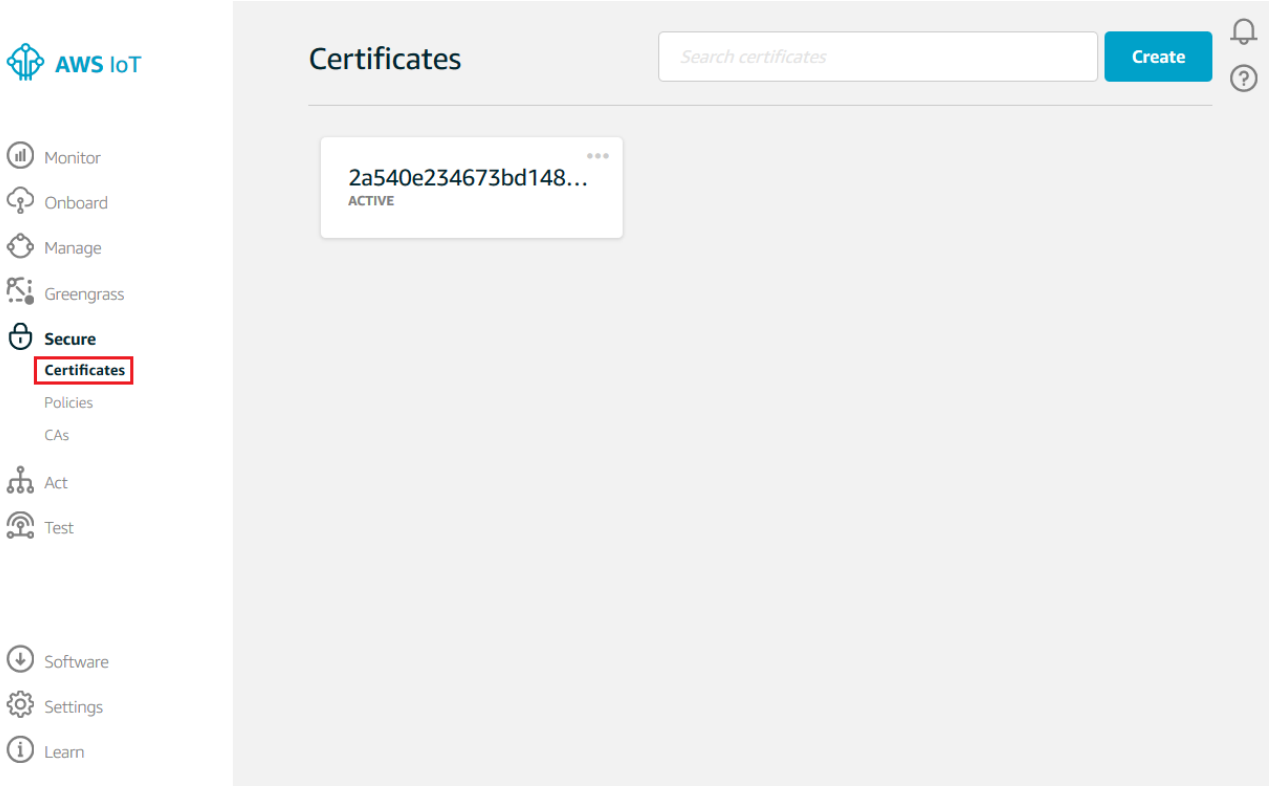
2. On the **Create a policy** page, in the **Name** field, enter a name for the policy (for example, MyIotPolicy). In the **Action** field, enter **iot:Connect**. In the **Resource ARN** field, enter *****. Select the **Allow** check box. This allows all clients to connect to AWS

IoT. After you have entered the information for your policy, choose **Create**.

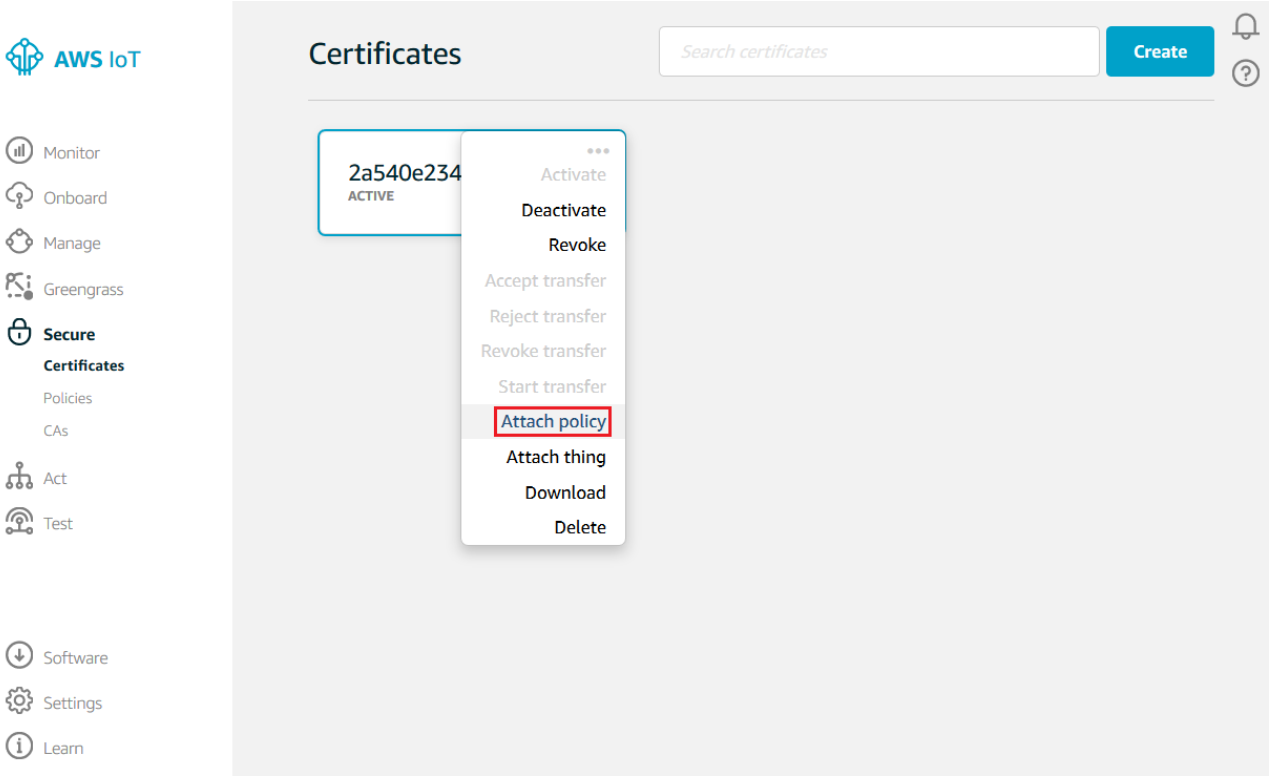


4.3 Attach an AWS IoT Policy to a Device Certificate

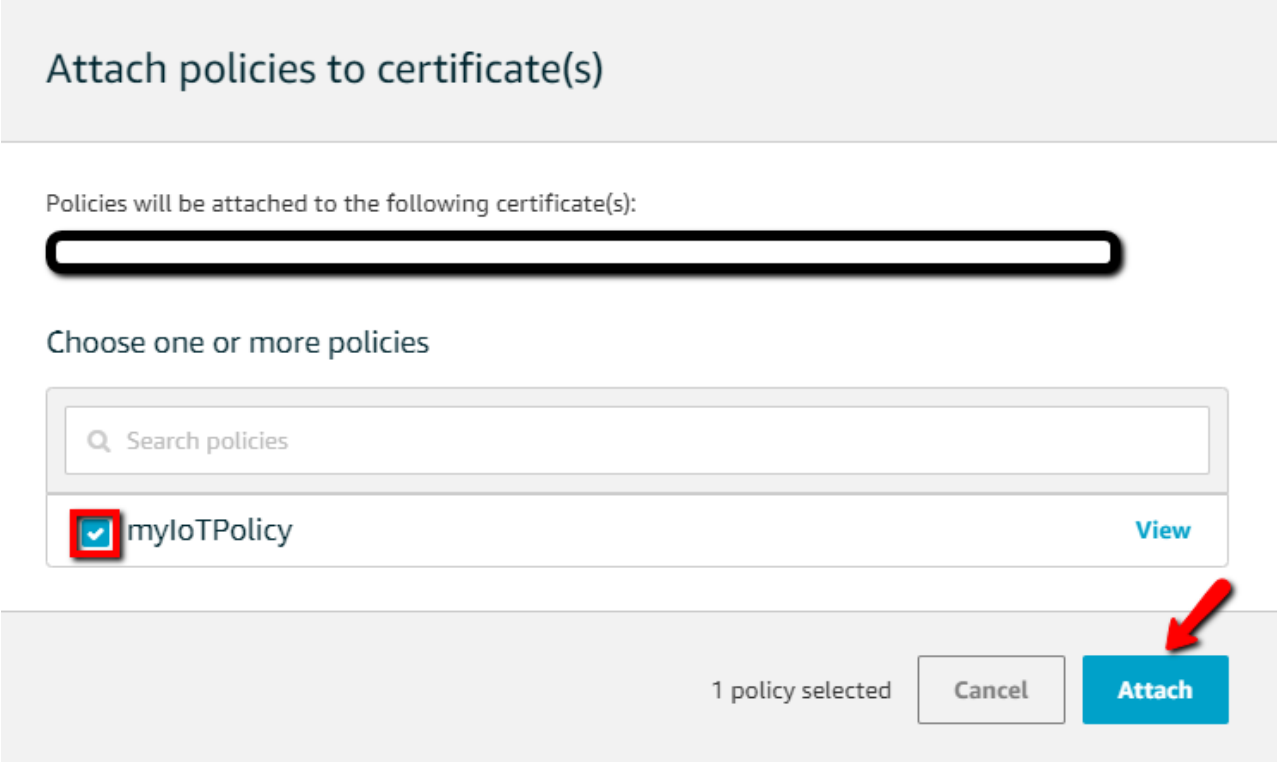
1. In the left navigation pane, choose **Secure**, and then choose **Certificates**.



2. In the box for the certificate you created, choose **...** to open a drop-down menu, and then choose **Attach policy**.

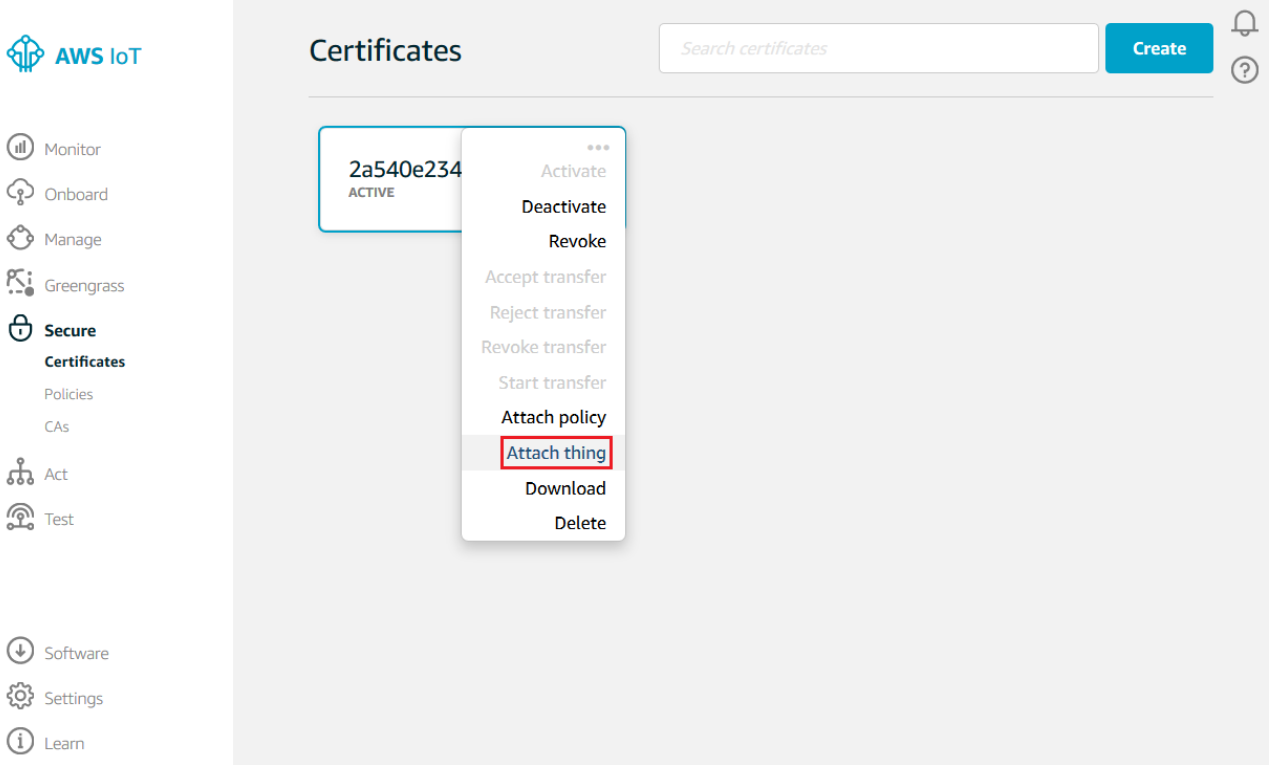


3. In **Attach policies to certificate(s)**, select the check box next to the policy you created in the previous step, and then choose **Attach**.

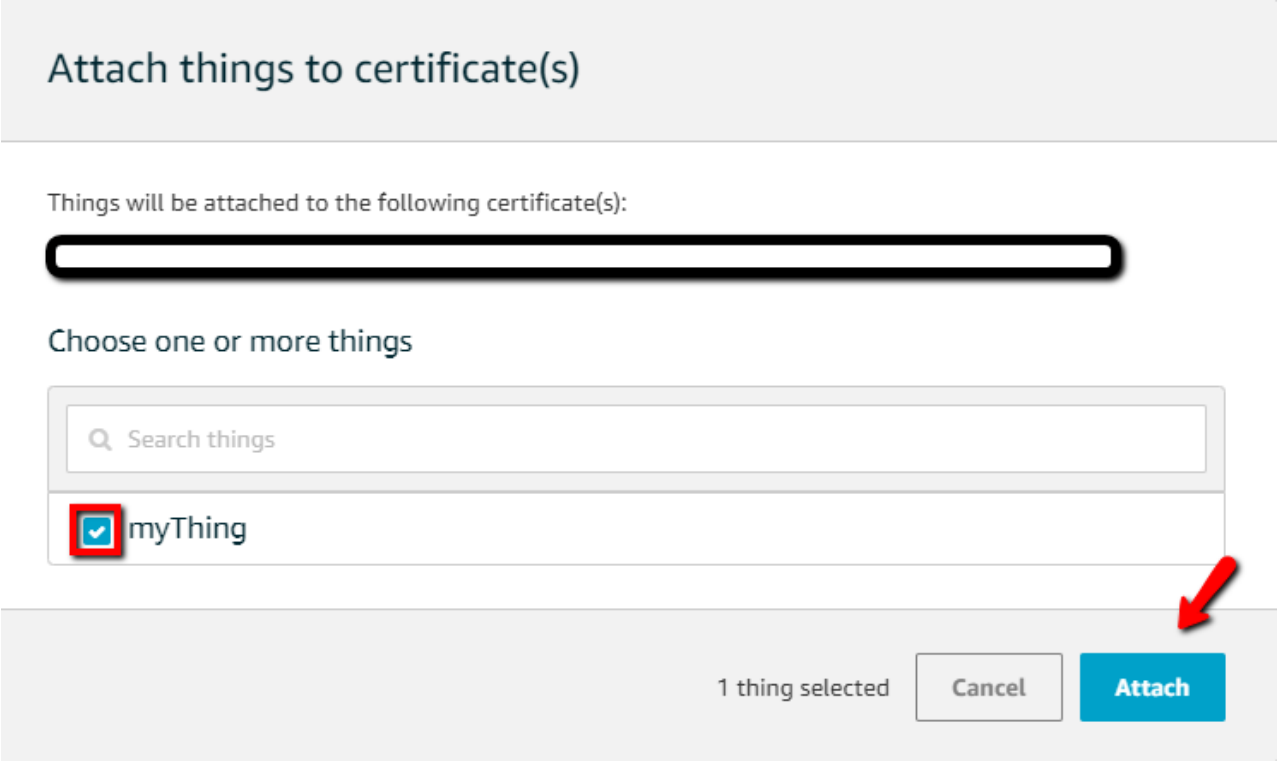


4.4 Attach a Certificate to a Thing

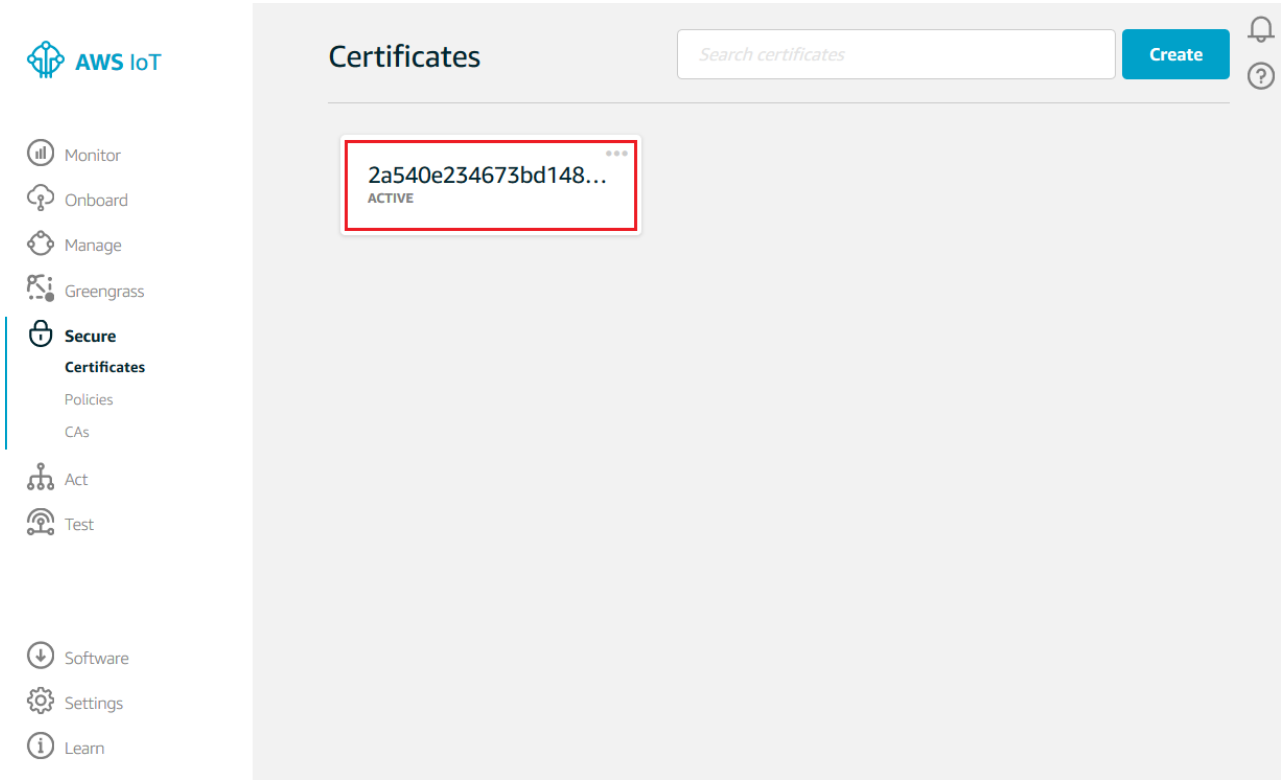
1. In the box for the certificate you created, choose **...** to open a drop-down menu, and then choose **Attach thing**.



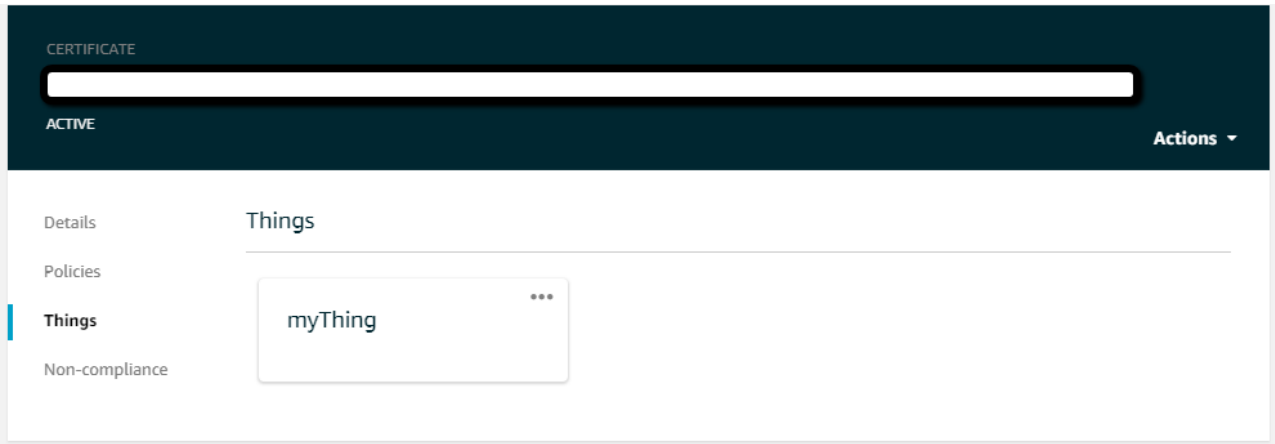
2. In **Attach things to certificate(s)**, select the check box next to the thing you registered, and then choose **Attach**.



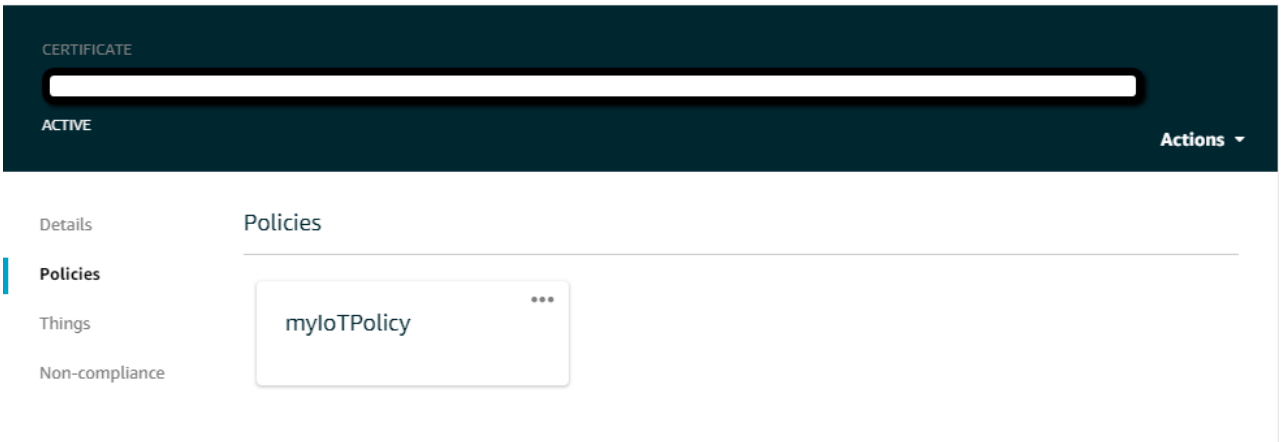
3. To verify the thing is attached, select the box for the certificate.



4. On the **Details** page for the certificate, in the left navigation pane, choose **Things**.



5. To verify the policy is attached, on the **Details** page for the certificate, in the left navigation pane, choose **Policies**.

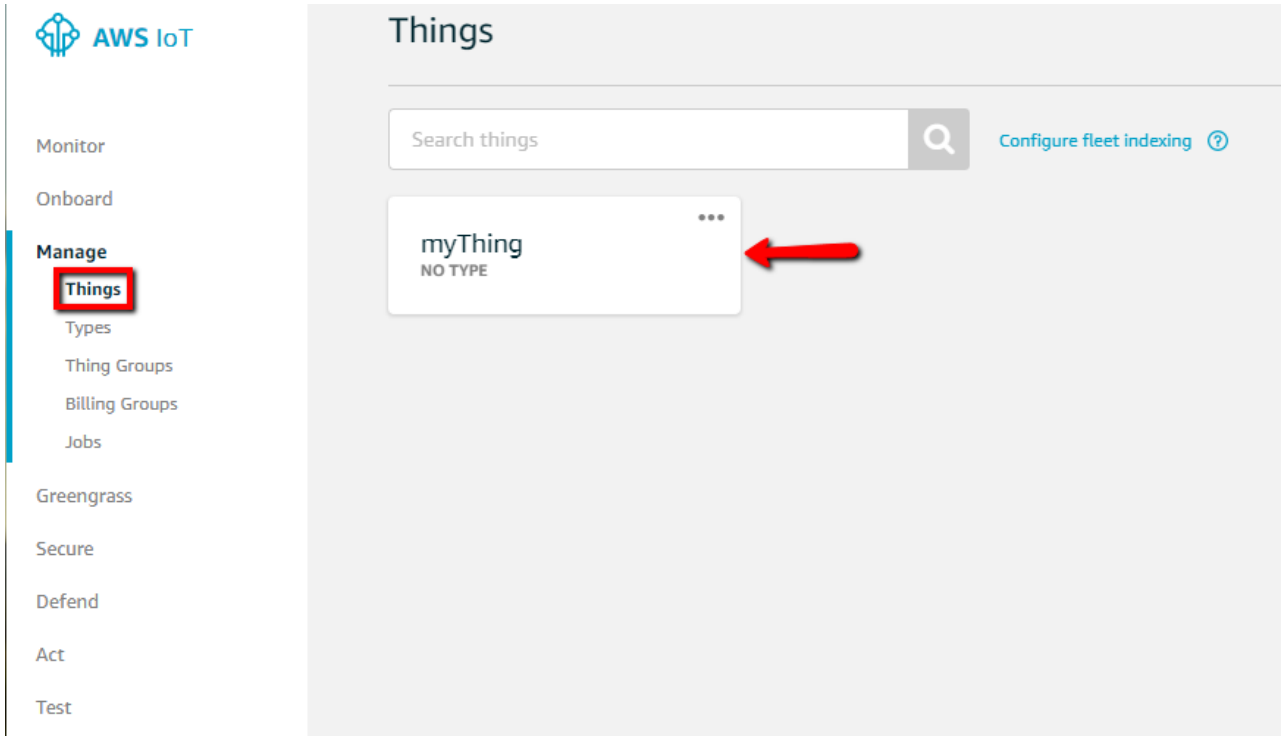


Chapter 5
Configure the device

The related SDK code folder is available here: `SDK_2.8.0_EVK-MIMXRT1060\boards\evkmimxrt1060\aws_examples\ota_demo_enet`.

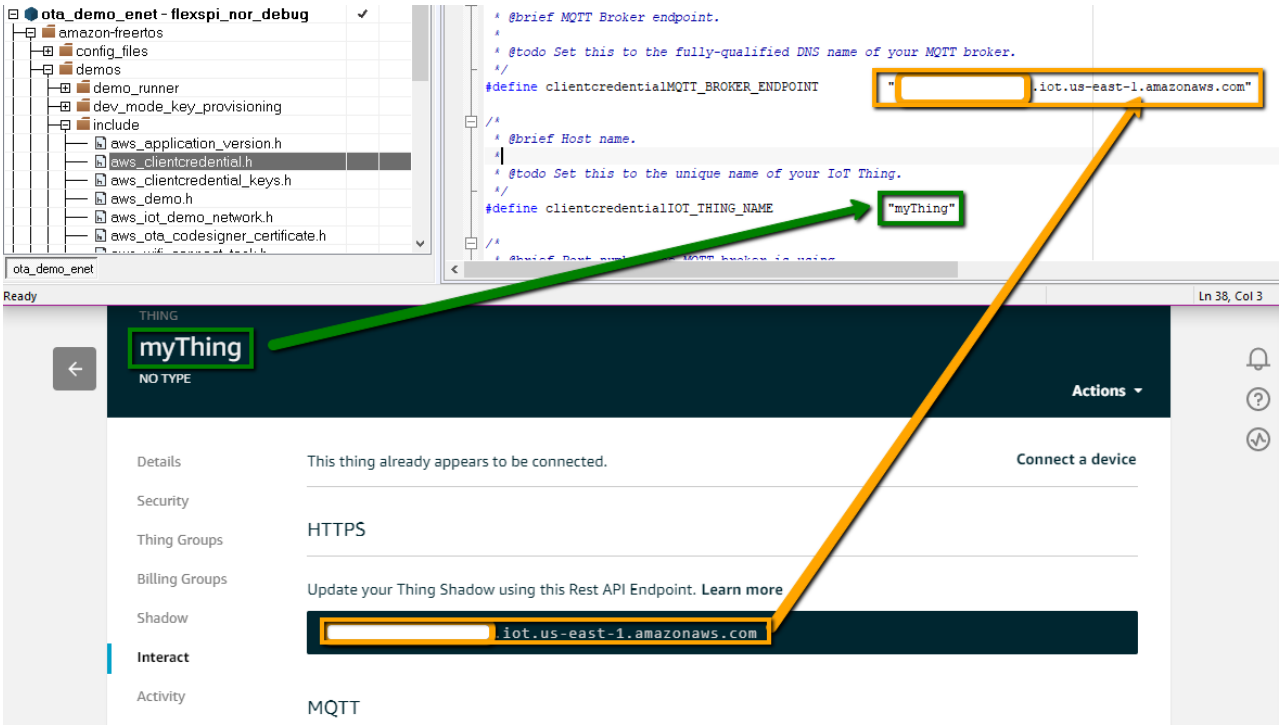
5.1 aws_clientcredential.h

1. Open the AWS IoT console website https://console.aws.amazon.com/iot/
2. On the **Welcome to the AWS IoT Console** page, in the navigation pane, choose **Manage – Things** select the previously created **Thing**.



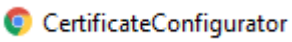
3. In the navigation pane, choose **Interact,** copy the **REST API** endpoint and **IoT Thing** name.

4. Inside the OTA project, open amazon-freertos – demos – include – aws_clientcredential.h and set the **REST_API** and **IoT Thing** name obtained in the previous step.

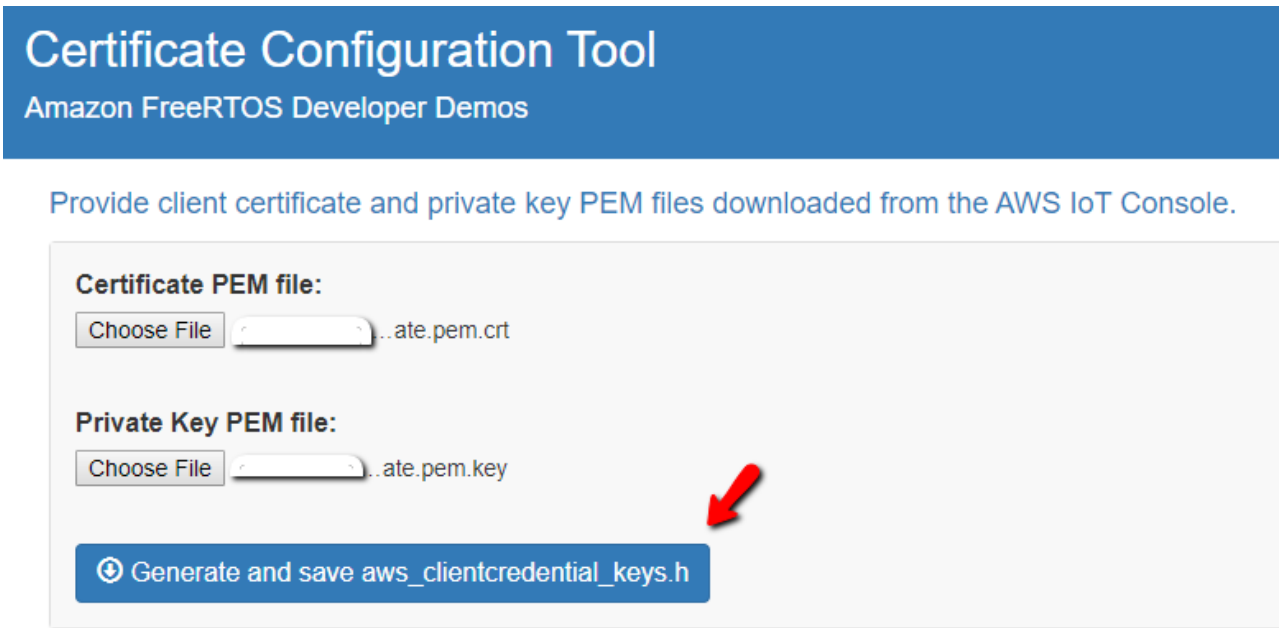


5.2 aws_clientcredential_keys.h

1. In your PC go to …\rtos\amazon-freertos\tools\certificate_configuration
2. Using a web browser open the CertificateConfigurator.html



3. Browse to the Certificate and Key files previously downloaded from your Thing and click on **Generate and save aws_clientcredential_keys.h**



4. Using windows explorer replace the…\rtos\amazon-freertos\demos\include\aws_clientcredential_keys.h with the file obtained in the previousy step.

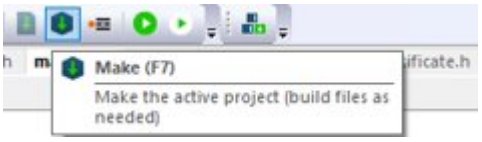
5.3 aws_ota_codesigner_certificate

1. Open the ecdsaigner.crt file using a text editor
2. Copy all the content – Paste the information at the OTA project amazon-freertos – demos – include – aws_ota_codesigner_certificate.h in the signingcredentialSIGNING_CERTIFICATE_PEM Note: be sure to add **"** at the beging of a line and **\n"** on every line break

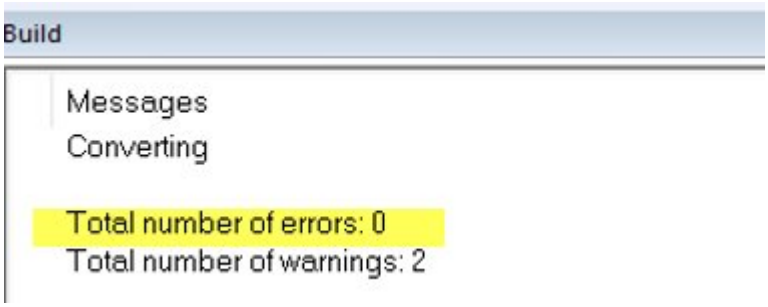
```
/*
 * PEM-encoded code signer certificate
 *
 * Must include the PEM header and footer:
 * "-----BEGIN CERTIFICATE-----\n"
 * "...base64 data...\n"
 * "-----END CERTIFICATE-----\n";
 */
static const char signingcredentialSIGNING_CERTIFICATE_PEM[] =
"-----BEGIN CERTIFICATE-----\n"
"MIIBYTCCAQegAwIBAgIJAKCX9bIhkilFMAoGCCqGSM49BAMCMCMxITAfBgNVBAMM\n"
"GEFsZWphbmRyYS5HdXptYW5AbnhwLmNvbTAeFw0xOTEwMjMxNjMzNDJaFw0yMDEw\n"
"MjIxNjMzNDJaMCMxITAfBgNVBAMMGEFsZWphbmRyYS5HdXptYW5AbnhwLmNvbTBZ\n"
"MBMGByqGSM49AgEGCCqGSM49AwEHA0IABGUghBD51mF1J3wf4LYsQ2VgOaDpg98G\n"
"dNC38FWGS7owT4NC5848JumrD8SonnnXpu77Pt7ShuW39hC3Vdi7z1GjJDAiMAsG\n"
"A1UdDwQEAwIHgDATBgNVHSUEDDAKBggrBgEFBQcDAzAKBggqhkjOPQQDAgNIADBF\n"
"AiEAr0pNz1aMax4arCPNiW9HYFdQTvUGyZdRLcDrUol/LQoCIH2U2REoZ59V7r6z\n"
"CMLfHA+kWq84IjxDUE2OgV60RVvC\n"
"-----END CERTIFICATE-----\n";
```

5.4 Build

1. Click the make button to start building the application



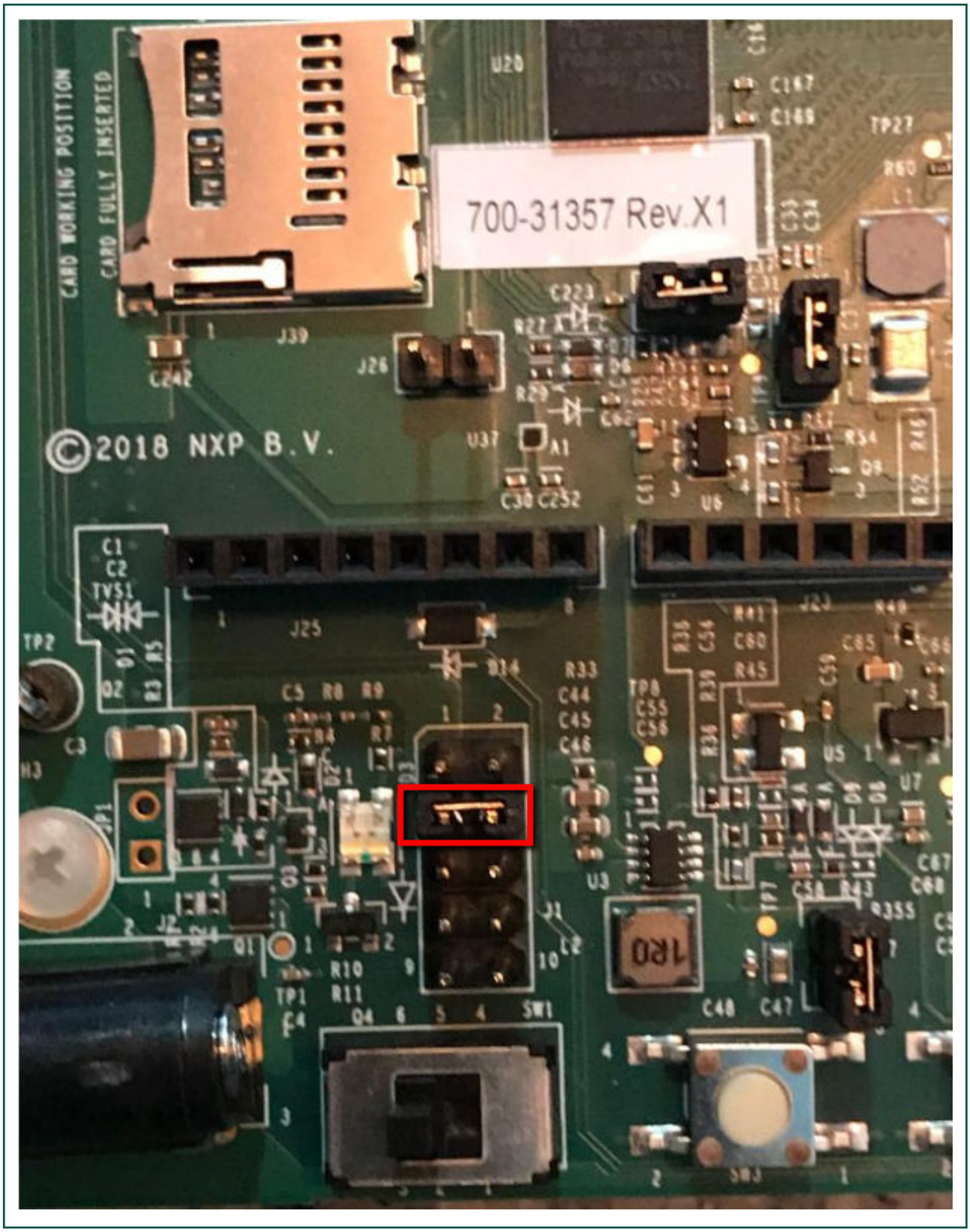
2. If build is successful Zero errors message is printed in build console



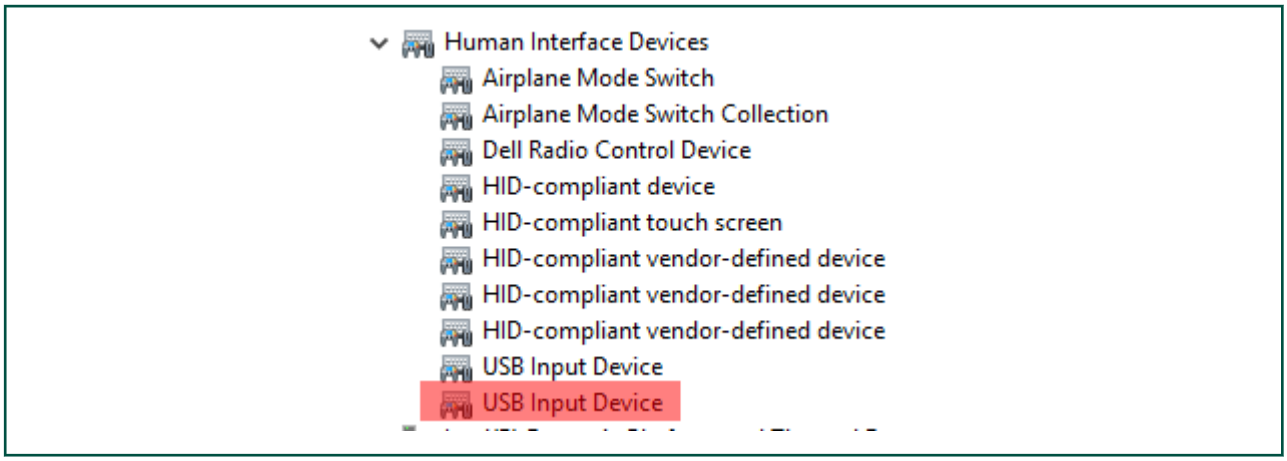
5.5 Programming mcu-boot into flash

1. Set all SW7 positions to off.

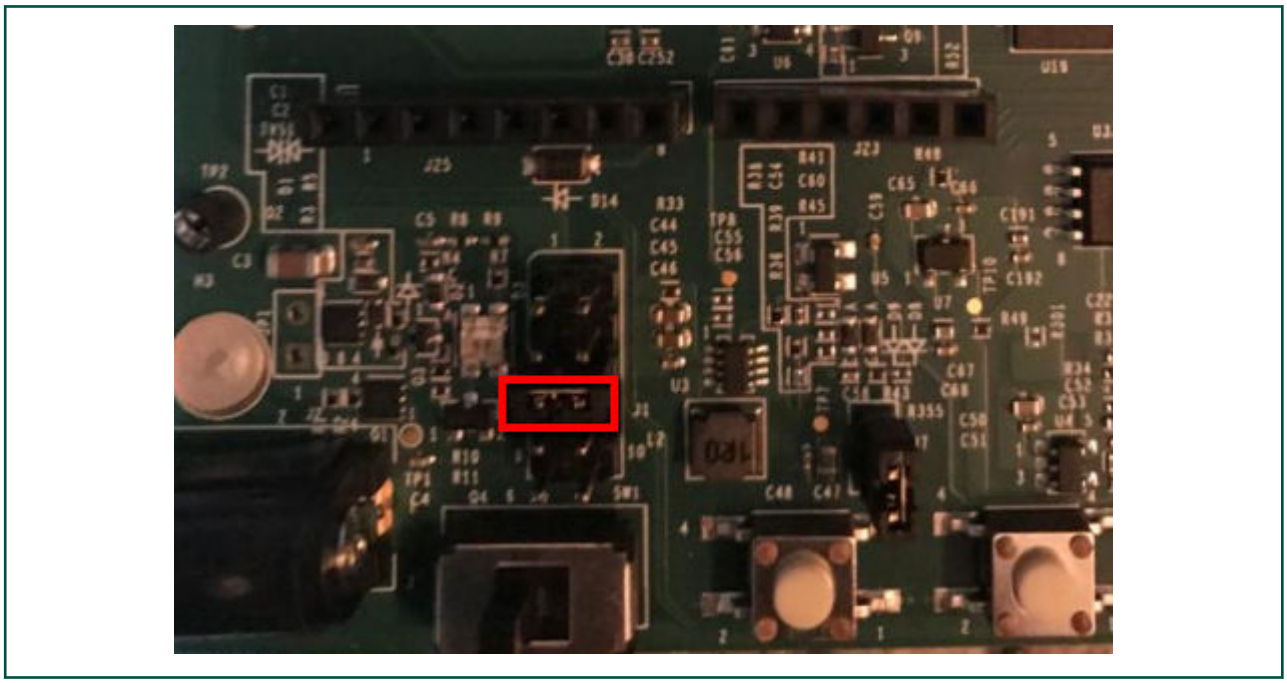
2. Locate J1, then move the jumper to **3-4**.



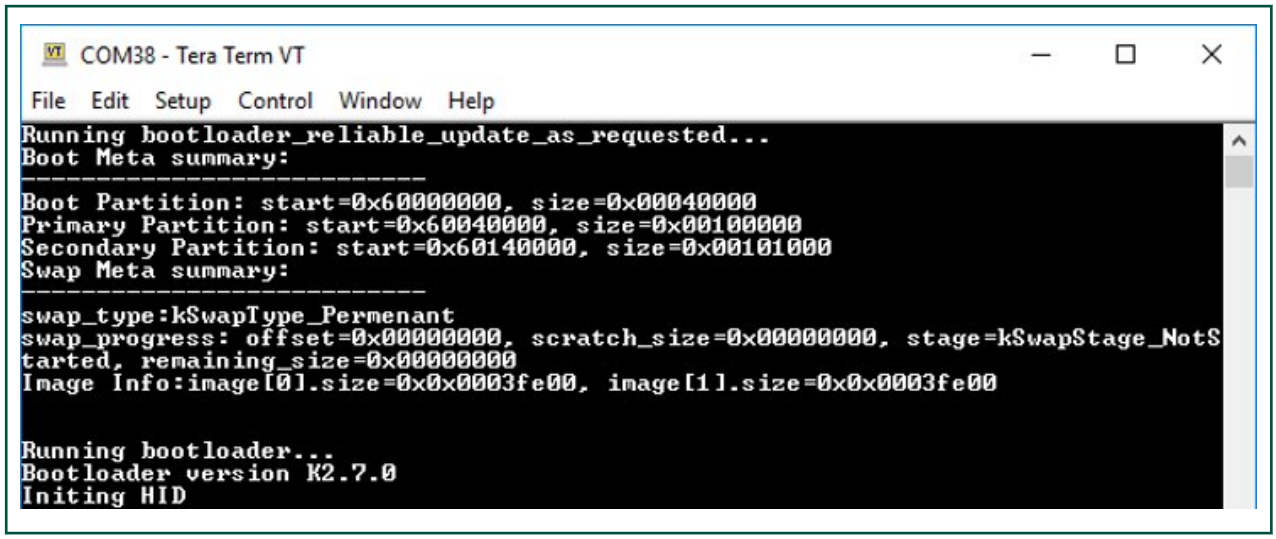
3. Connect the board to PC via J9 USB connector
4. Reset the board using **SW3**, then make sure that your RT1060-EVK gets enumerated like Human Iinterface Devices – USB Input device



5. Open a windows Command Prompt then execute the following commands It is recommended (but not required) to have bash interpreter at hand, git bash would do the job https://gitforwindows.org/ > cd ..\OTA_Bootloader_Scripts-4e081f \OTA_Bootloader_Scripts_0.5 > generate_ota_bootloader_and_program_it_to_flash.sh.
6. Disconnect the USB cable from the J9 USB connector.
7. Set SWD7[1:4]:0010.
8. Return J1 jumper to the default setting **5-6**.

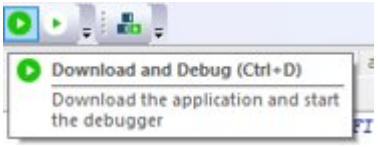


9. Connect the RT1060-EVK to the PC using the OpenSDA USB connector J41, mimxrt1060-evk, SCH rev A2, and use some terminal application to connect to the virtual com port so that you could to see the console
10. Reset the EVK using SW3 at this moment you should be able to see bootloader messages being printed on a terminal



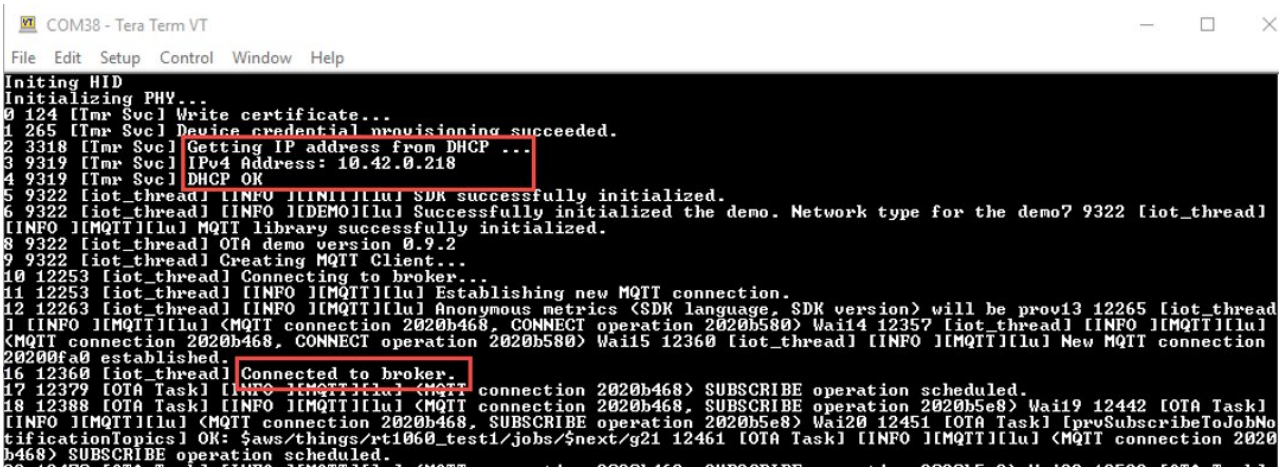
5.6 Flashing the OTA Agent application

1. Attach ethernet cable with Internet connection and local DHCP server
2. Click the download and debug button to start flashing the device



3. When the device is flashed, the debug pointer will highlight green the main entry point
4. Click the Go button to start running the program

Double check that there are MQTT AWS messages on the terminal

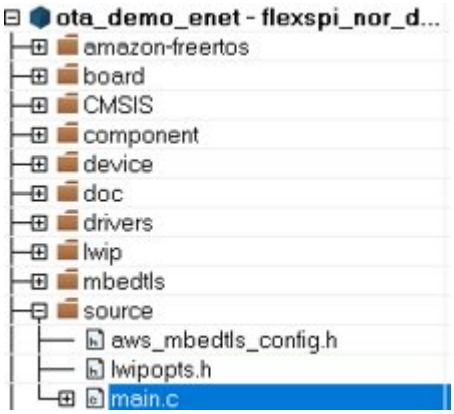


5. Stop the debug session

Chapter 6
OTA

6.1 Create new image

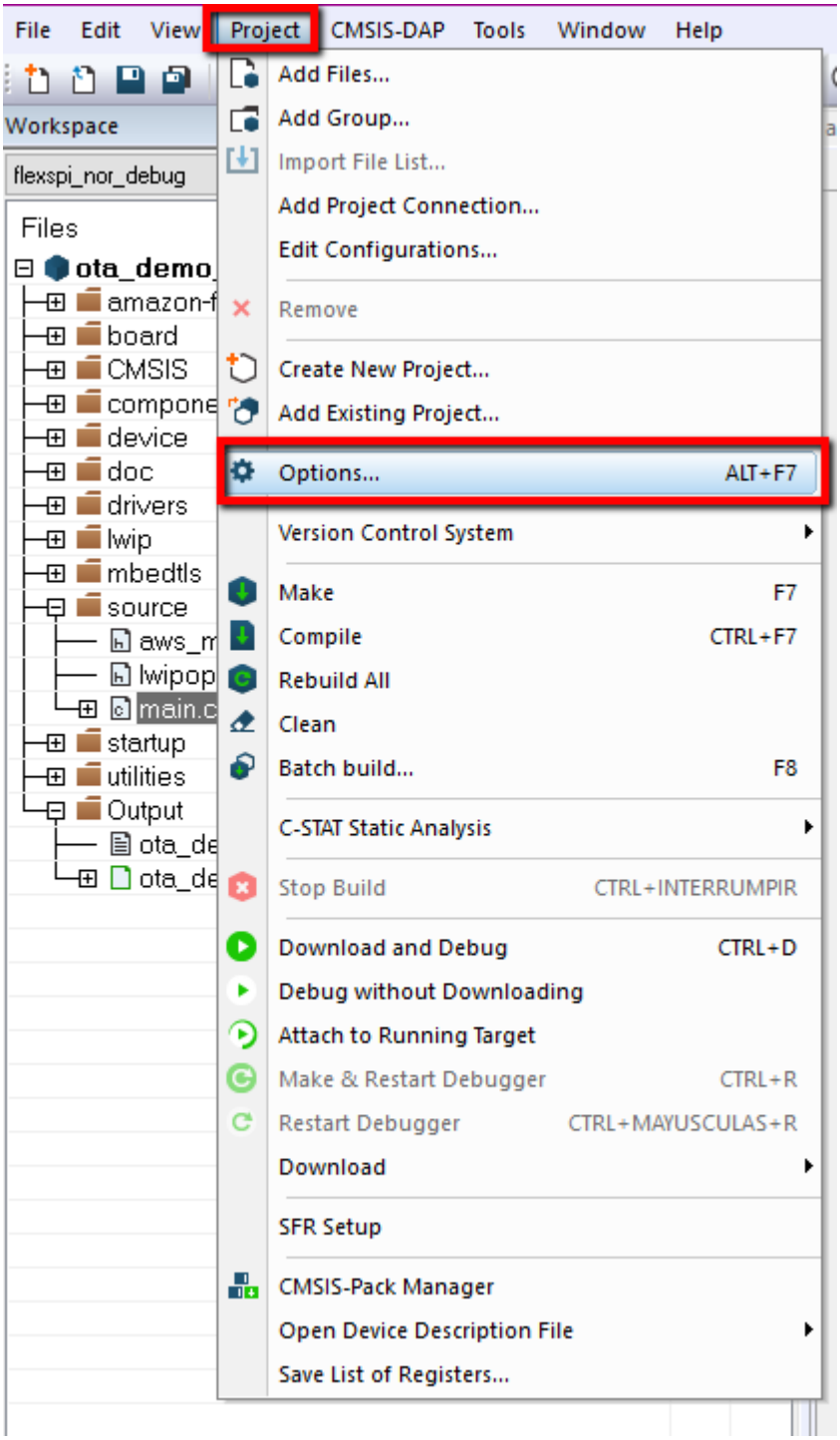
1. Open main.c



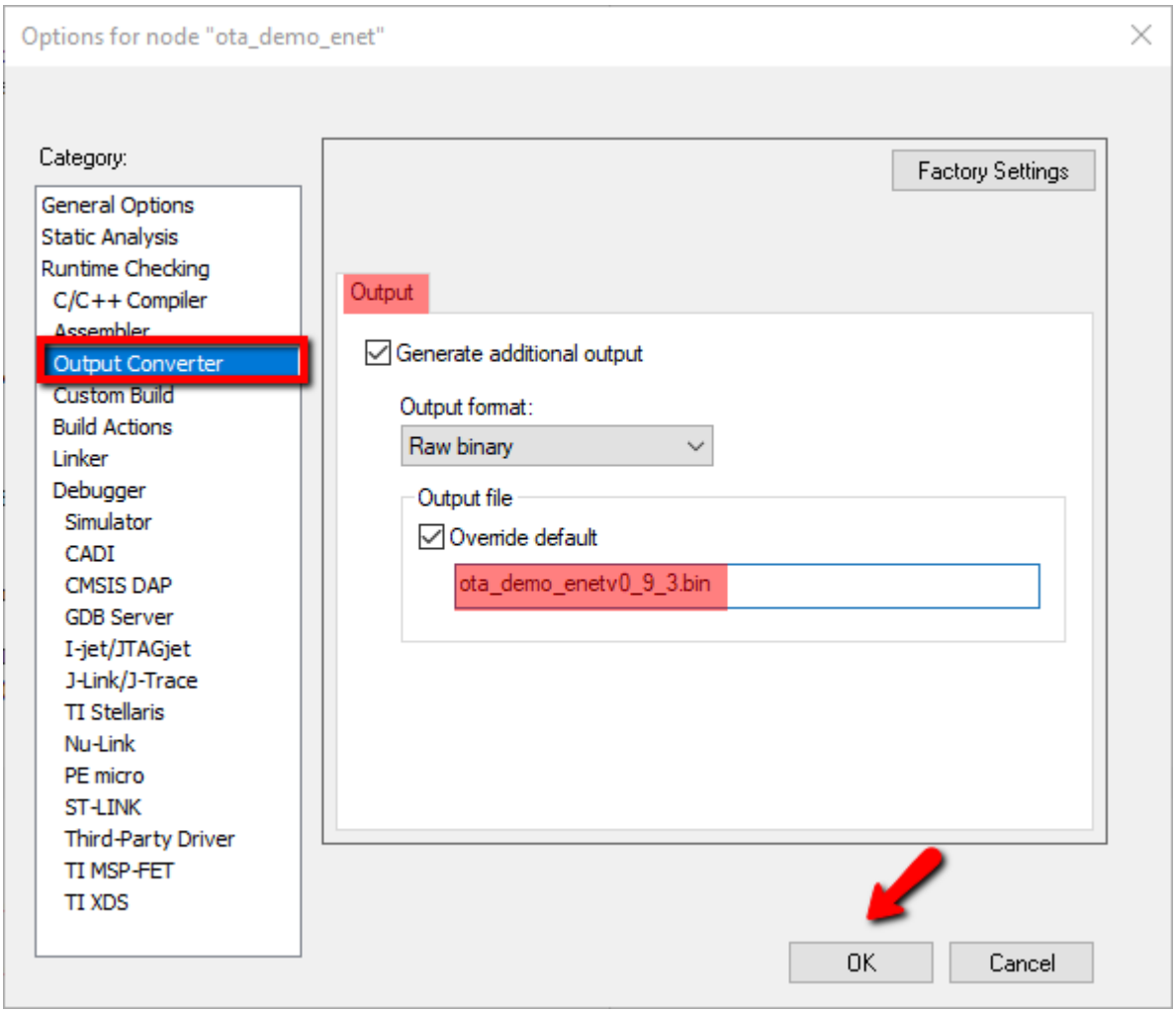
2. Go to line 86, then change any of the APP_VERSION macro to a higher number.

```
#define APP_VERSION_MAJOR 0
#define APP_VERSION_MINOR 9
#define APP_VERSION_BUILD 3//2
```

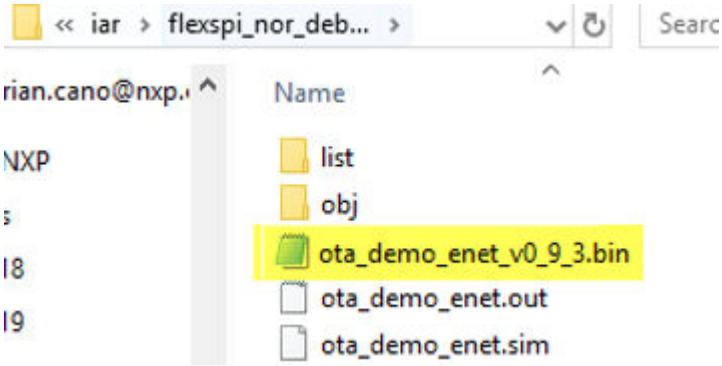
3. Open Project ☐ Options…



4. In the Category section choose **Output Converter** then change the name of the binary so it matches the version change then click **OK**.



5. Use the make button to build and generate the binary. Look for the binary inside the …boards \evkmimxrt1060\aws_examples\ota_demo_enet\iar\flexspi_nor_debug folder

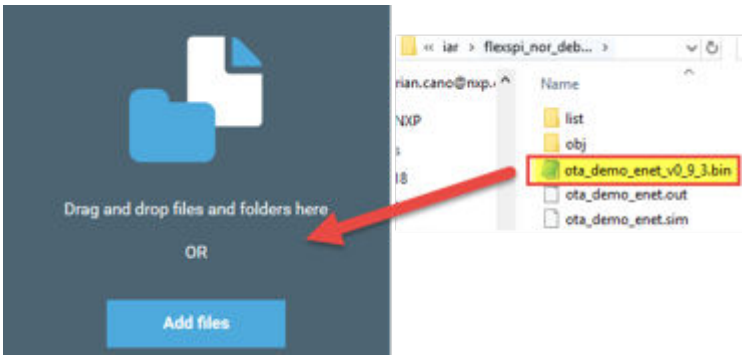


6.2 Uploading the binary to the S3 bucket

1. Use AWS console to open the S3 service https://console.aws.amazon.com/s3
2. Select the previously created bucket

3. Click **Upload**

4. Drag and drop the ota_demo_enet_v0_9_3.bin binary



5. Click **Upload**

6.3 Create OTA Job

1. Open the AWS IoT console website https://console.aws.amazon.com/iot/
2. On the **Welcome to the AWS IoT Console** page, in the navigation pane, choose **Manage – Jobs**.

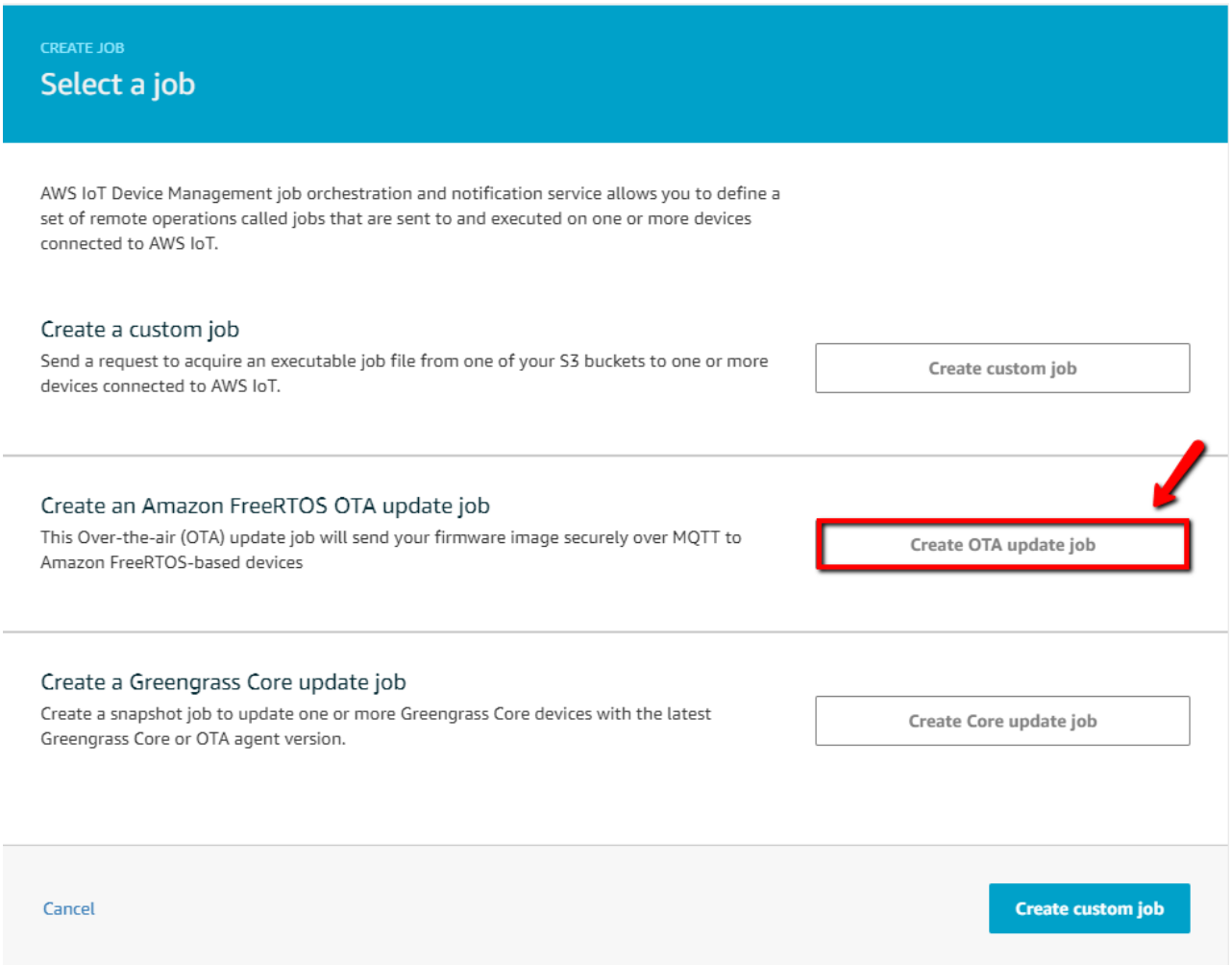
3. Select **Create**

Start a job for your devices

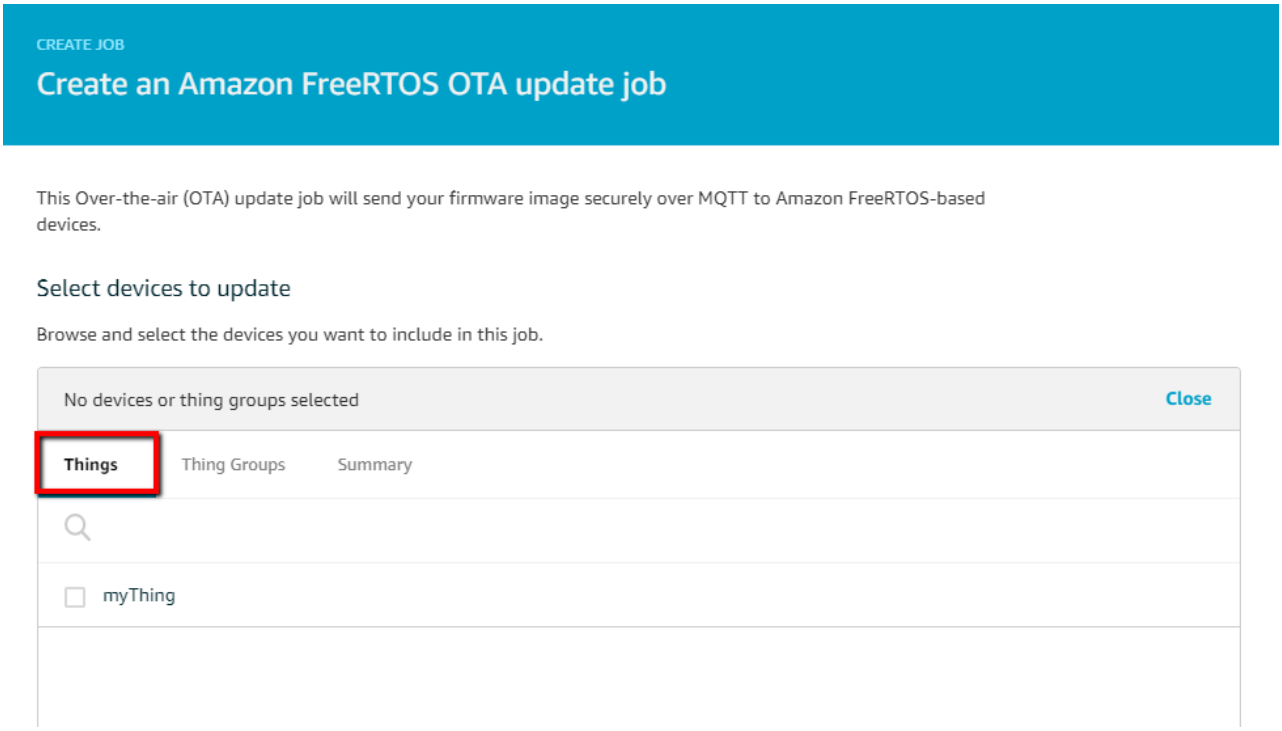
AWS IoT Device Management allows you to send files or deployments to devices.

Learn more | Create a job

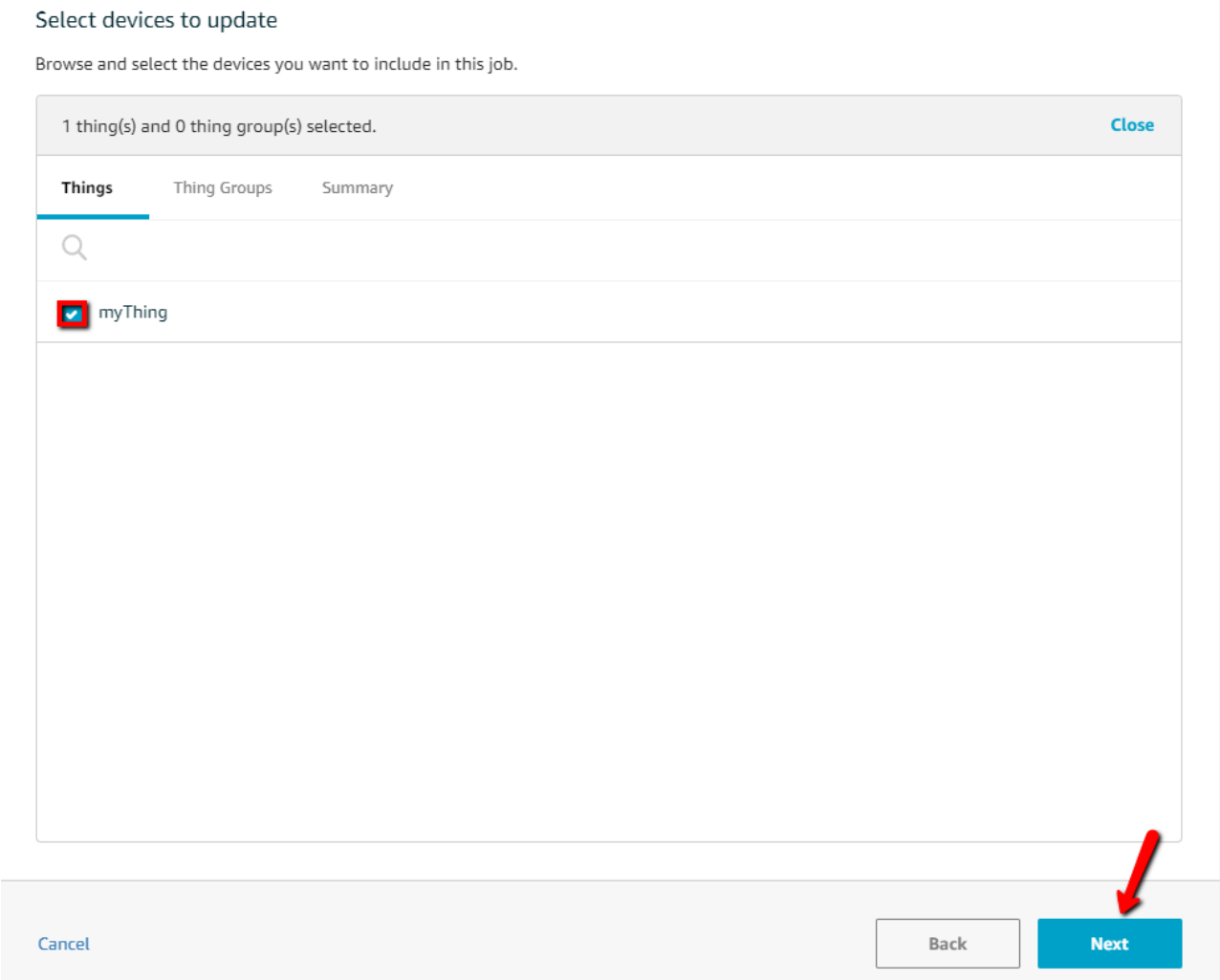
4. Under **Create an Amazon FreeRTOS Over-the-Air (OTA) update job**, choose **Create OTA update job**.



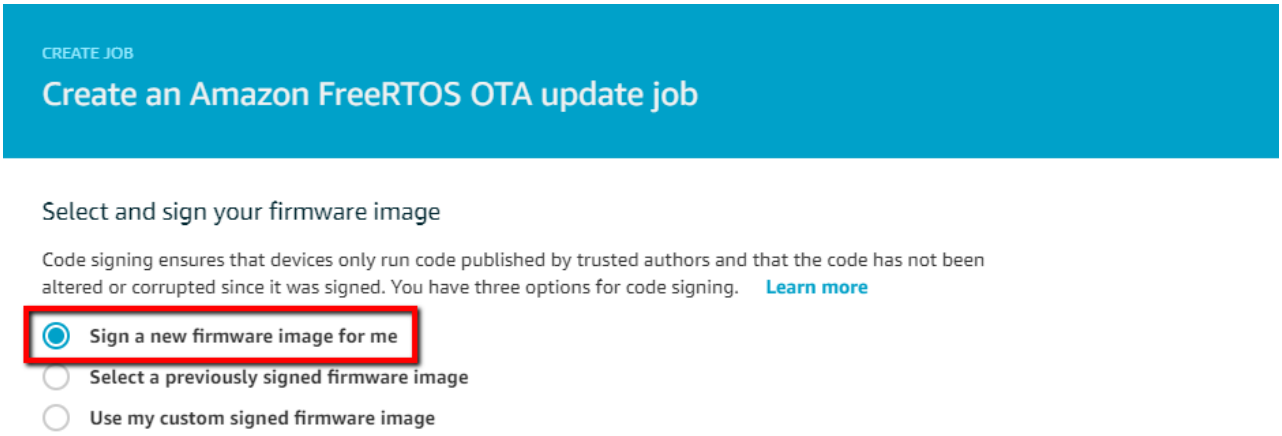
5. Under **Select devices to update**, choose **Select**. To update a single device, choose the **Things** tab



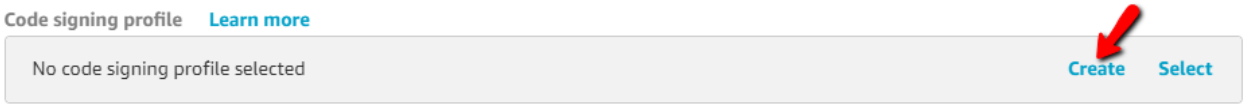
6. Select the check box next to the IoT thing associated with your device. Choose **Next**.



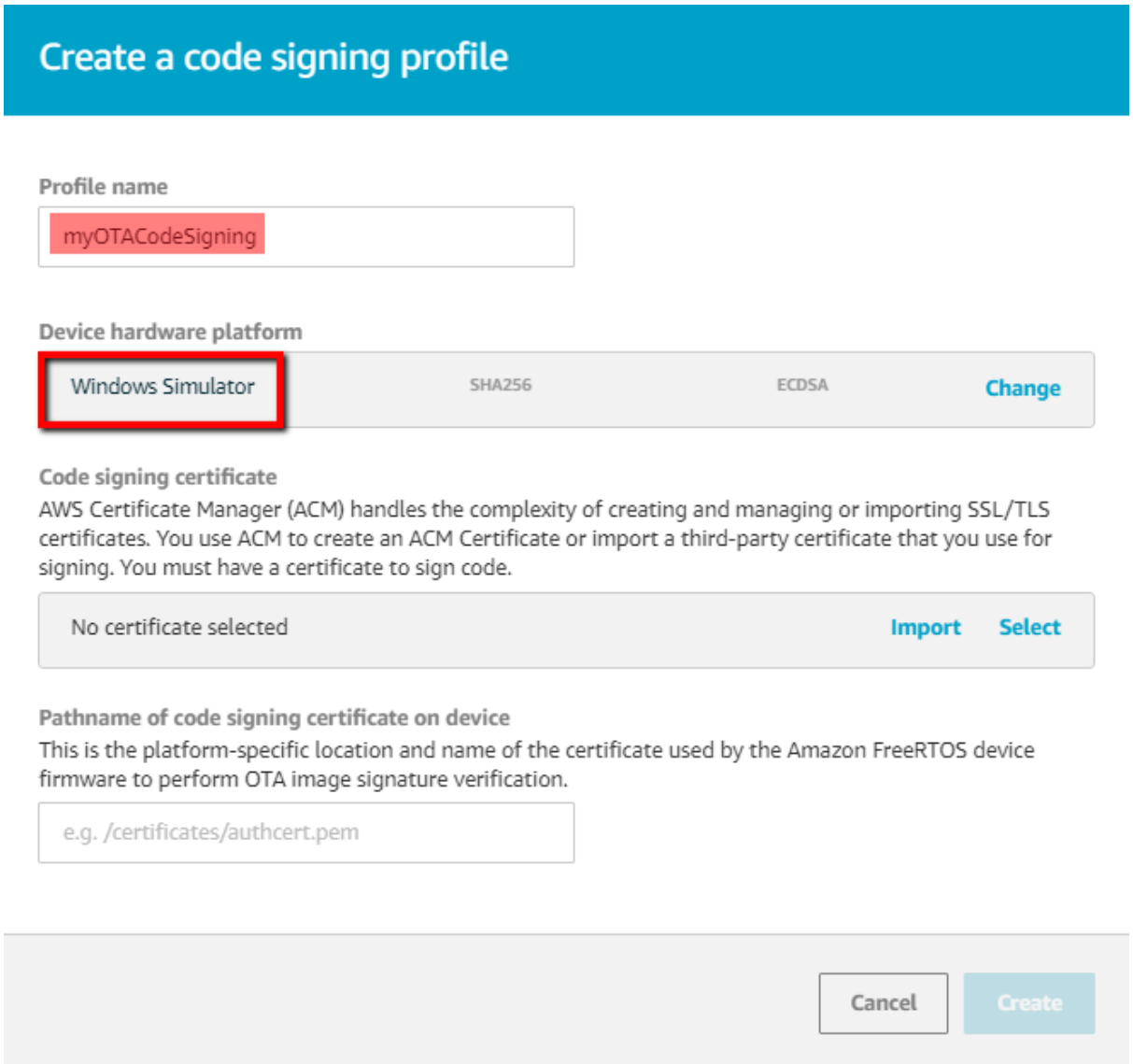
7. Under **Select and sign your firmware image**, choose **Sign a new firmware image for me**.



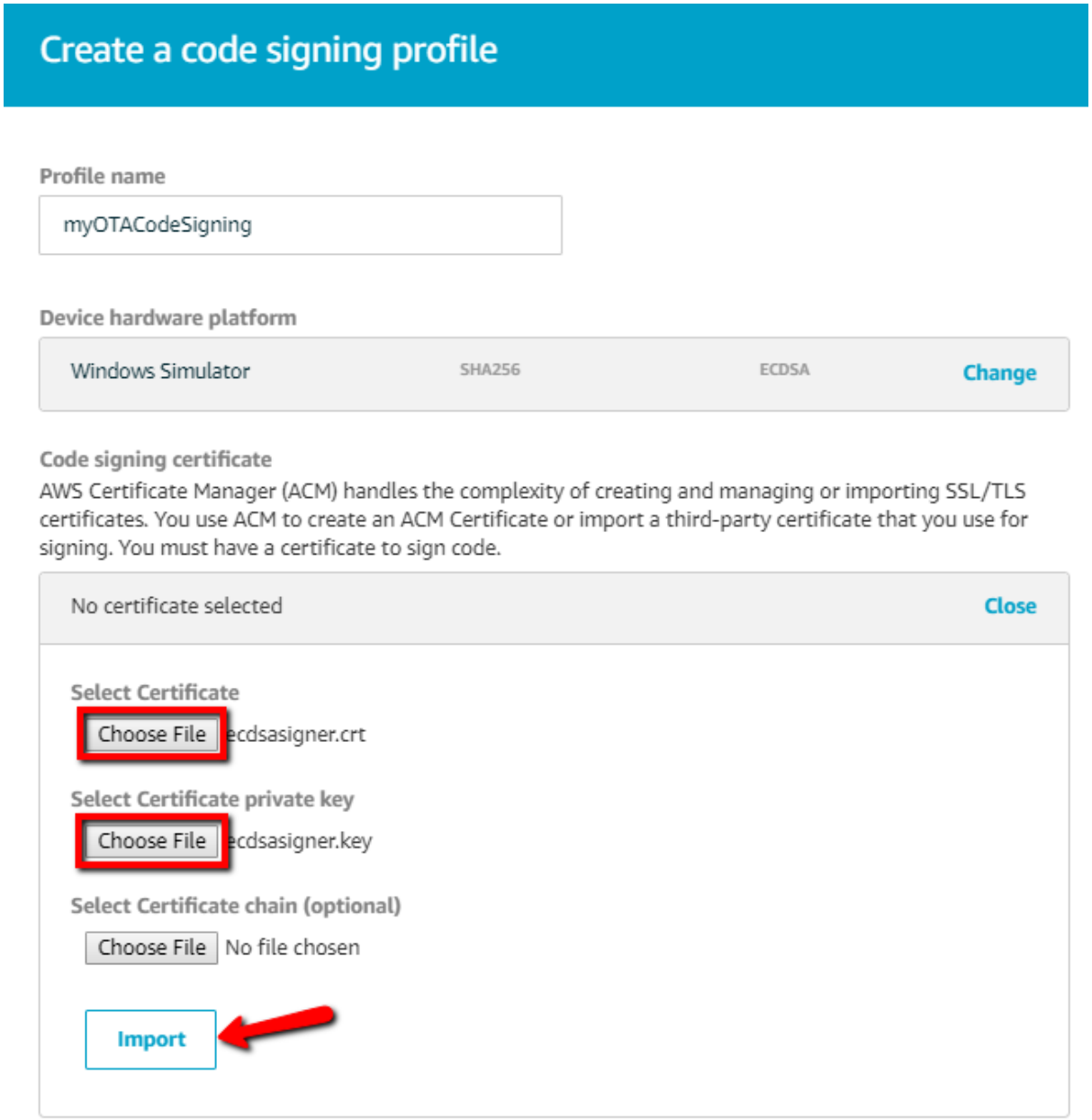
8. Under **Code signing profile**, choose **Create**.



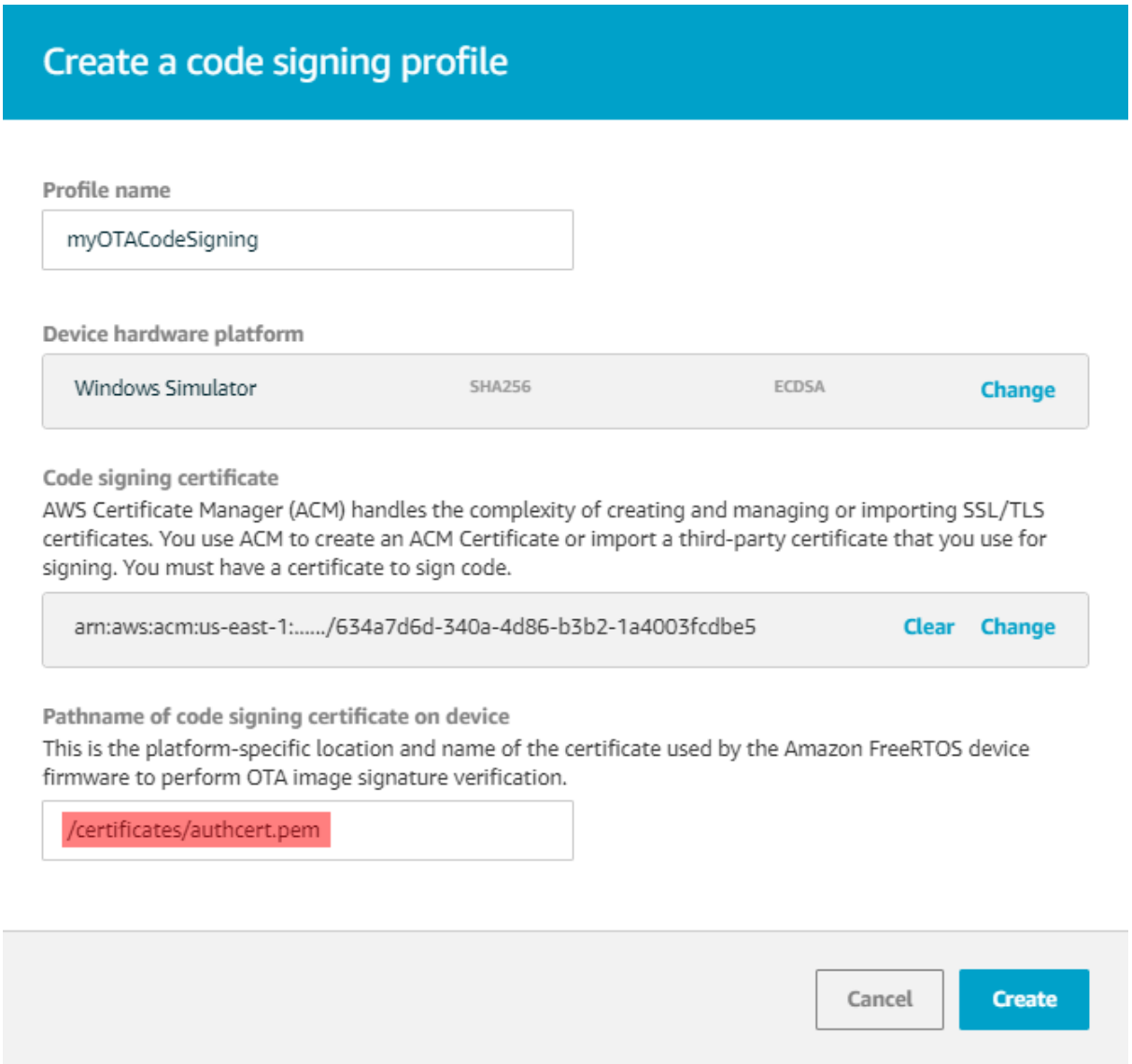
9. In **Create a code signing profile**, enter a name for your code-signing profile.
   a. Under **Device hardware platform**, choose Windows Simulator.



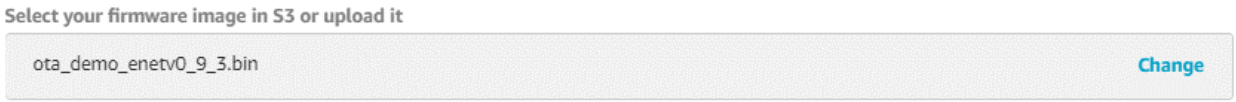
b. Under **Code signing certificate**, choose **Import** and browse for the ecda certificate created with AWS CLI.



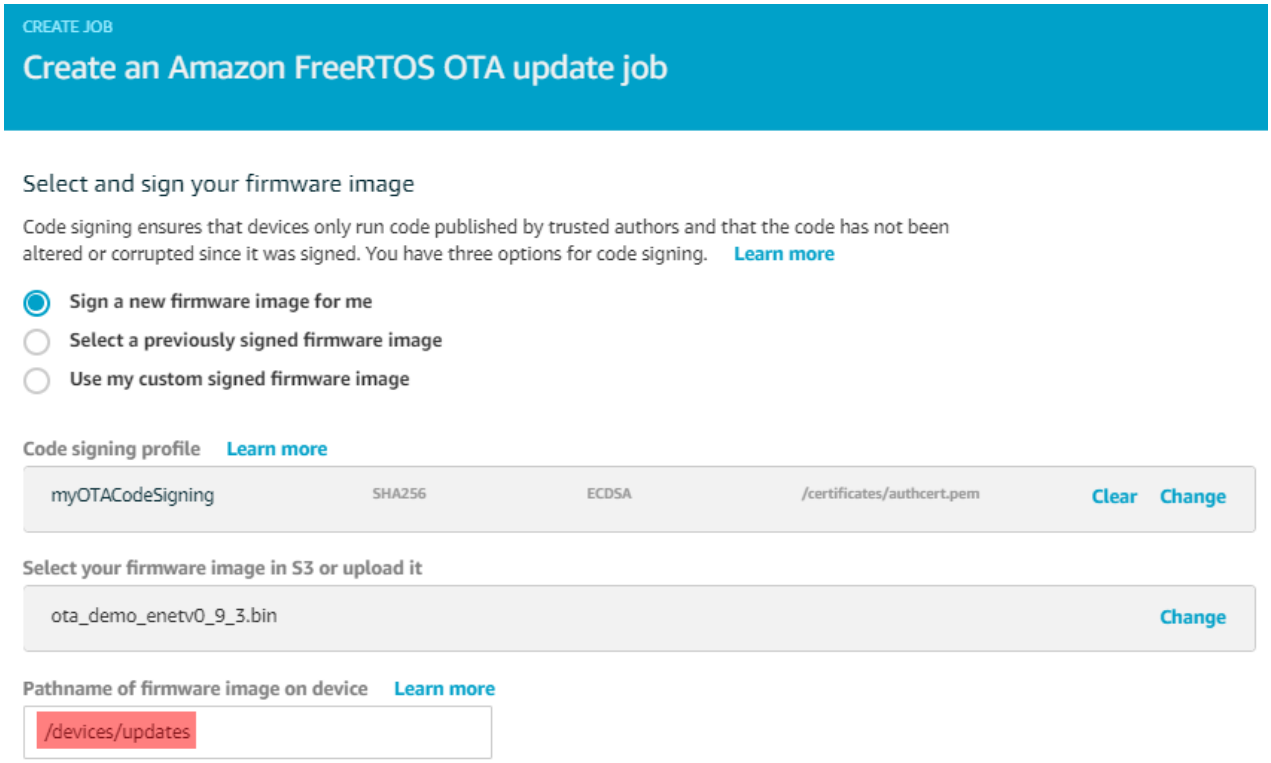
c. Under **Pathname of code signing certificate on device**, type the default path /certificates/authcert.pem then click **Create**.



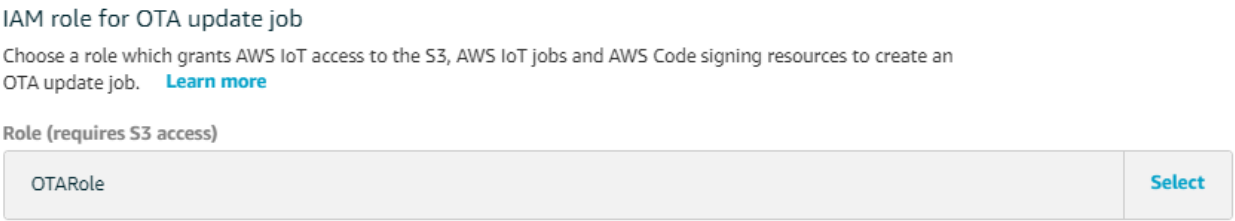
10. Under **Select your firmware image in S3**, choose **Select**.



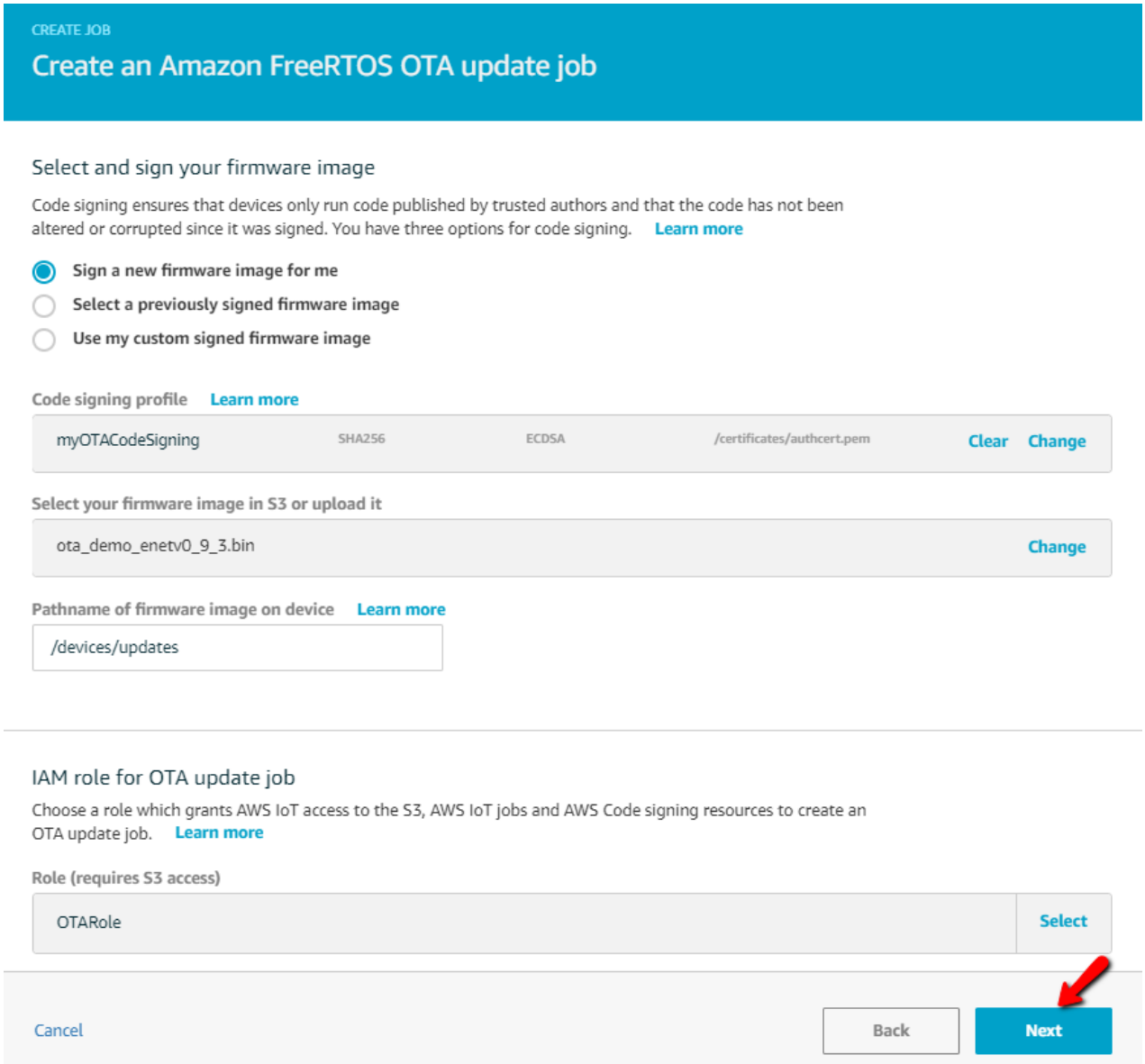
11. Under **Pathname of firmware image on device**, type the default path /device/updates



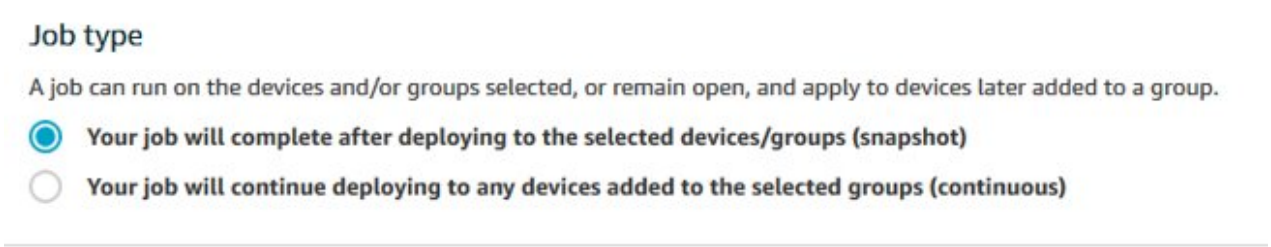
12. Under **IAM role for OTA update job**, choose the role created in previous steps.



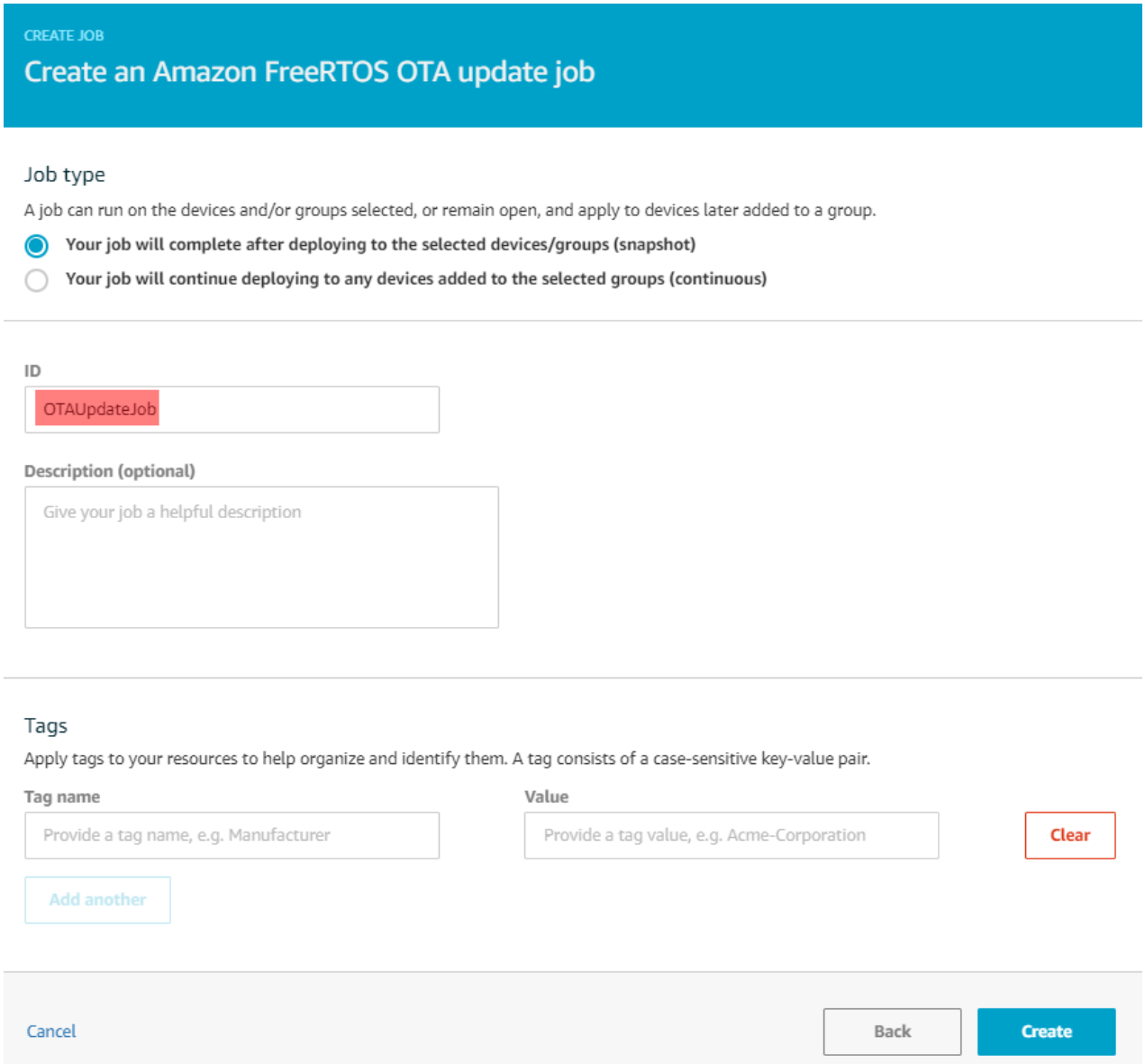
13. Choose **Next**.



14. Under **Job type**, choose **Your job will complete after deploying to the selected devices/groups (snapshot)**.



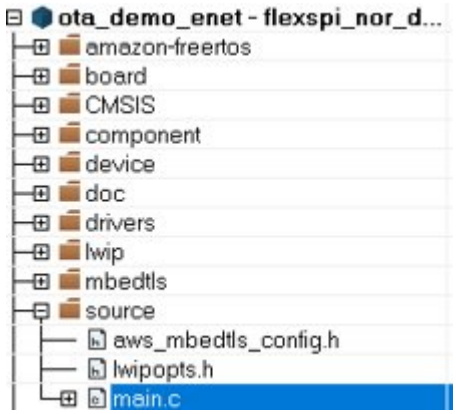
15. Enter an ID for your OTA update job



16. Before clicking Create the application needs to run

6.4 Running the application

1. Open main.c.



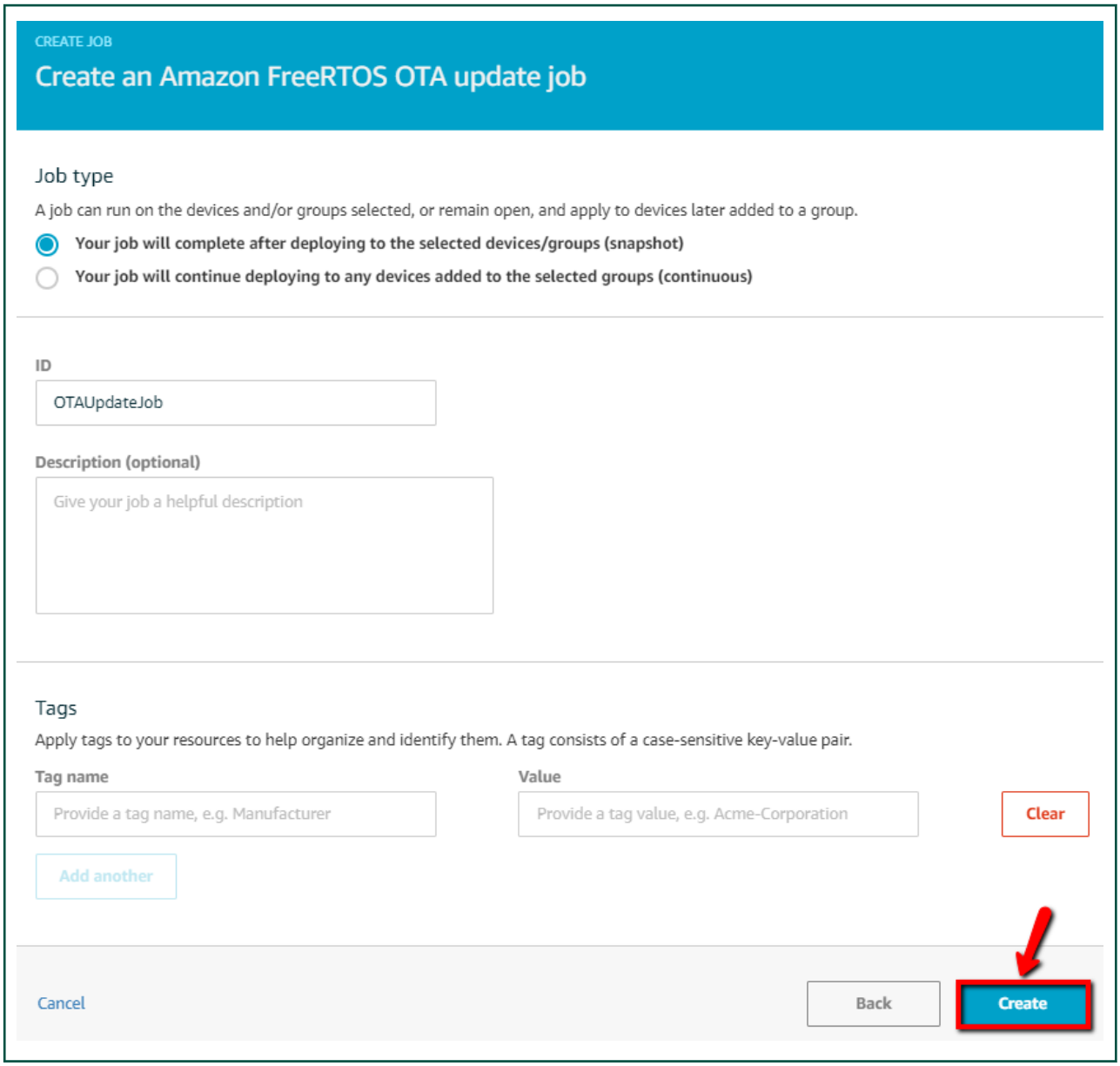
2. Go to line 86, then change any of the APP_VERSION macro to the original value.

```
#define APP_VERSION_MAJOR 0
#define APP_VERSION_MINOR 9
#define APP_VERSION_BUILD 2
```

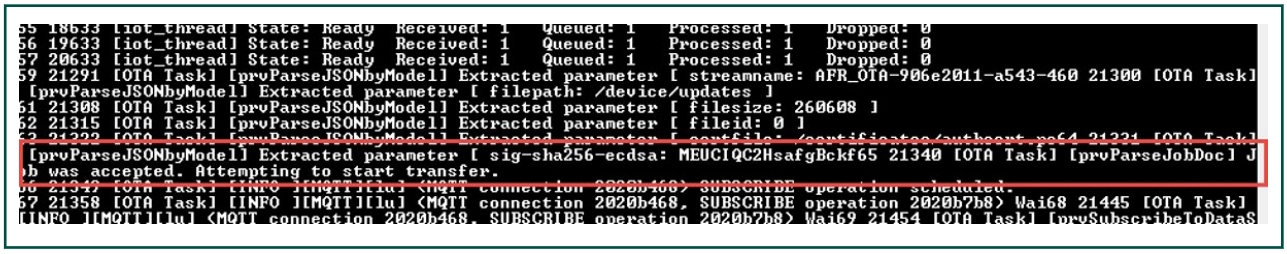
3. Make and Download & Debug.
4. When running the application wait until the message of the OTA State Ready is shown in the serial terminal.

```
60 23602 [iot_thread] State: Ready  Received: 1   Queued: 1   Processed: 1   Dropped: 0
61 24602 [iot_thread] State: Ready  Received: 1   Queued: 1   Processed: 1   Dropped: 0
62 25602 [iot_thread] State: Ready  Received: 1   Queued: 1   Processed: 1   Dropped: 0
63 26602 [iot_thread] State: Ready  Received: 1   Queued: 1   Processed: 1   Dropped: 0
64 27602 [iot_thread] State: Ready  Received: 1   Queued: 1   Processed: 1   Dropped: 0
```

5. At this point the OTA agent is waiting for an OTA job. Go back to the Create OTA job window and click **Create**.



6. The process will start, you can see a similar output.



7. Start file transfer.

```
928 42555 [OTA Task] [OTA-NXP] WriteBlock 3dc00 : 400
929 42560 [OTA Task] [prvIngestDataBlock] Remaining: 2
930 42634 [iot_thread] State: Active  Received: 317   Queued: 255   Processed: 255   Dropped: 62
931 43634 [iot_thread] State: Active  Received: 317   Queued: 255   Processed: 255   Dropped: 62
932 44634 [iot_thread] State: Active  Received: 317   Queued: 255   Processed: 255   Dropped: 62
933 45048 [OTA Task] [INFO ][MQTT][lu] (MQTT connection 2020b468) MQTT PUBLISH operation queued.
934 45057 [OTA Task] [prvPublishGetStreamMessage] OK: $aws/things/rt1060_test1/streams/AFR_OTA-906
] [prvIngestDataBlock] Received file block 242, size 1024
936 45256 [OTA Task] [OTA-NXP] WriteBlock 3c800 : 400
937 45261 [OTA Task] [prvIngestDataBlock] Remaining: 1
938 45266 [OTA Task] [prvIngestDataBlock] Received file block 252, size 1024
939 45273 [OTA Task] [OTA-NXP] WriteBlock 3f000 : 400
940 45278 [OTA Task] [prvIngestDataBlock] Received final expected block of file.
```

8. Swap.

```
--------------------------
Swap is in progress...
swap_type:kSwapType_Test
swap_progress: offset=0x00000000, scratch_size=0x00000000, stage=kSwapStage_Done, remaining_size=0x00000000
Image Info:image[0].size=0x0x0003fe00, image[1].size=0x0x0003fe00
```

9. Device gets restarted, then the new application starts running.

```
Running bootloader...
Bootloader version K2.7.0
Initing HID
Initializing PHY...
0 124 [Tmr Svc] Write certificate...
1 266 [Tmr Svc] Device credential provisioning succeeded.
2 1946 [Tmr Svc] Getting IP address from DHCP ...
3 4946 [Tmr Svc] IPv4 Address: 10.42.0.218
4 4946 [Tmr Svc] DHCP OK
5 4949 [iot_thread] [INFO ][INIT][lu] SDK successfully initialized.
6 4949 [iot_thread] [INFO ][DEMO][lu] Successfully initialized the de
[INFO ][MQTT][lu] MQTT library successfully initialized.
8 4949 [iot_thread] OTA demo version 0.9.3
9 4949 [iot_thread] Creating MQTT Client...
```

Chapter 7
Revision history

This table summarizes revisions to this document.

Table 1. Revision history

Revision number	Date	Substantive changes
0	12/2019	Initial release
1	06/2020	Updated for MCUXpresso SDK v2.8.0

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

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

Date of release: 15 June 2020
Document identifier: AWSOTAUG