

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



# Contents

<b>Chapter 1 Overview.....</b>	<b>3</b>
<b>Chapter 2 AWS OTA Pre-Requisites.....</b>	<b>4</b>
2.1 Create an Amazon S3 Bucket to store your update.....	4
2.2 Create an OTA Update Service Role.....	6
2.3 Create an OTA User Policy.....	17
2.4 Windows Pre-Requisites.....	20
2.5 Creating a Code-Signing Certificate.....	23
<b>Chapter 3 Grant access to code signing for AWS IoT.....</b>	<b>25</b>
<b>Chapter 4 AWS IoT.....</b>	<b>29</b>
4.1 Create an AWS IoT Thing.....	29
4.2 Create an AWS IoT Policy.....	34
4.3 Attach an AWS IoT Policy to a Device Certificate.....	36
4.4 Attach a Certificate to a Thing.....	37
<b>Chapter 5 Configure the device.....</b>	<b>40</b>
5.1 aws_clientcredential.h.....	40
5.2 aws_clientcredential_keys.h.....	41
5.3 aws_ota_codesigner_certificate.....	42
5.4 Build.....	42
5.5 Programming mcu-boot into flash.....	43
5.6 Flashing the OTA Agent application.....	46
<b>Chapter 6 OTA.....</b>	<b>47</b>
6.1 Create new image.....	47
6.2 Uploading the binary to the S3 bucket.....	49
6.3 Create OTA Job.....	50
6.4 Running the application.....	60
<b>Chapter 7 Revision history.....</b>	<b>64</b>

# Chapter 1

## Overview

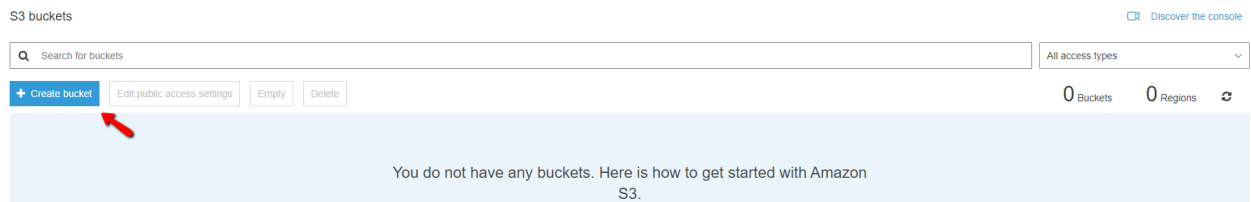
This guide walks through the steps to configure AWS services to make an Amazon FreeRTOS Over The Air Update using NXP's RT1060-EVK SDK. First, it creates an IAM role with OTA update, S3, IoT policies, and permissions. Then, using OpenSSL and AWS CLI commands, a code signing certificate is issued. Finally,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

# Chapter 2

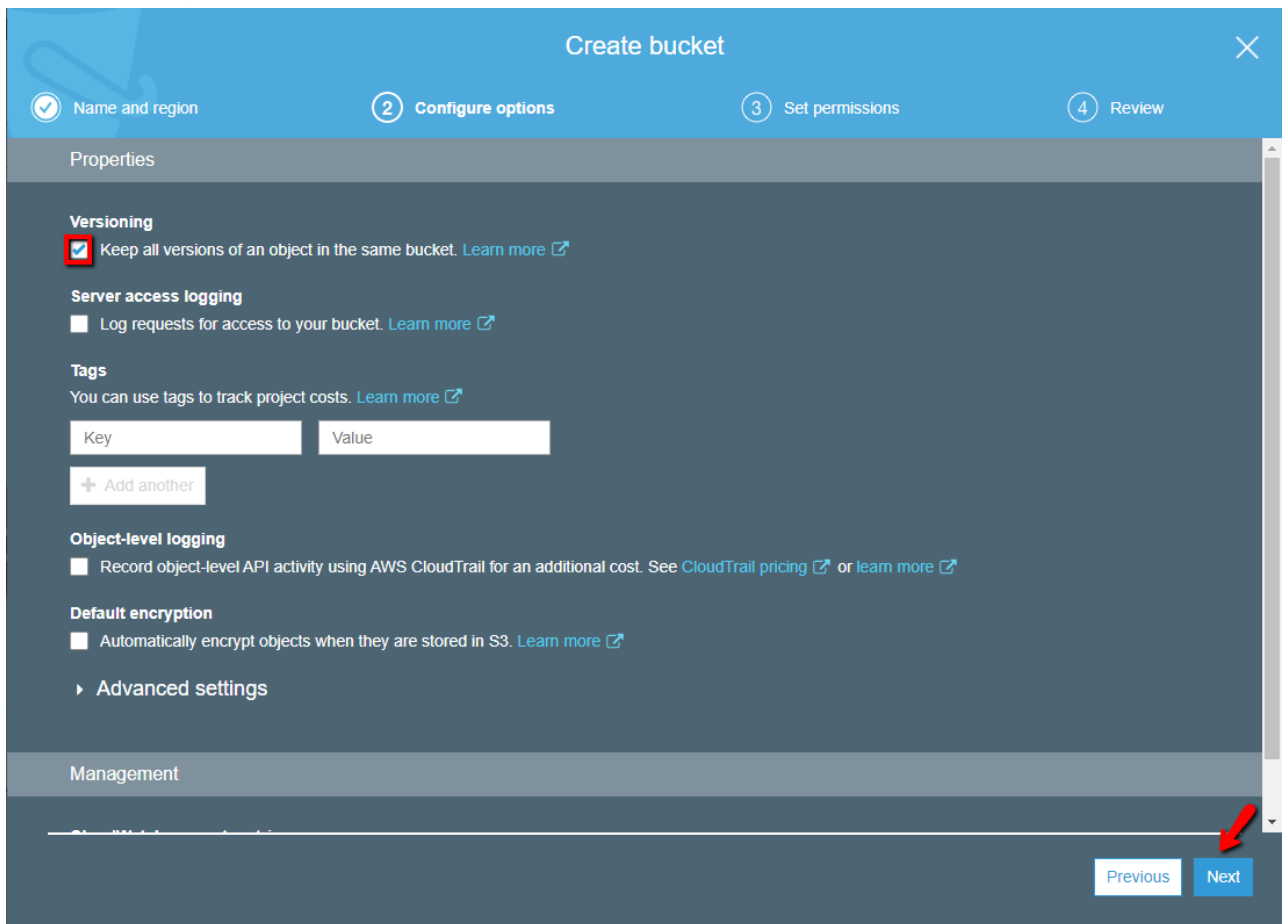
## AWS OTA Pre-Requisites

### 2.1 Create an Amazon S3 Bucket to store your update

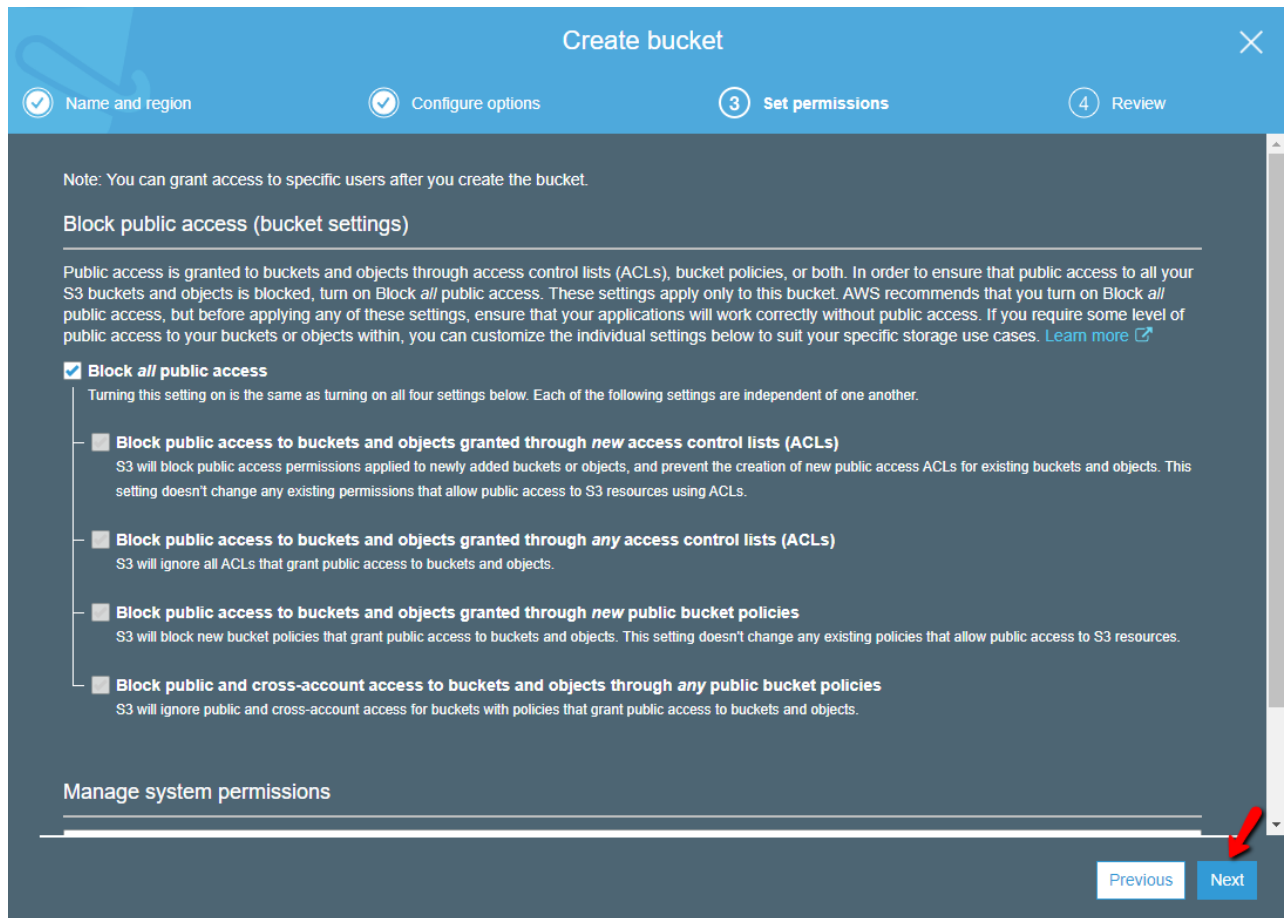
1. Go to the <https://console.aws.amazon.com/s3/>.
2. Choose **Create bucket**.



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.



**Create bucket**

1 Name and region 2 Configure options 3 **Set permissions** 4 Review

Note: You can grant access to specific users after you create the bucket.

### Block public access (bucket settings)

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, or both. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block *all* public access. These settings apply only to this bucket. AWS recommends that you turn on Block *all* public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

☒ **Block all public access**

Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

- ☐ **Block public access to buckets and objects granted through new access control lists (ACLs)**  
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- ☐ **Block public access to buckets and objects granted through any access control lists (ACLs)**  
S3 will ignore all ACLs that grant public access to buckets and objects.
- ☐ **Block public access to buckets and objects granted through new public bucket policies**  
S3 will block new bucket policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
- ☐ **Block public and cross-account access to buckets and objects through any public bucket policies**  
S3 will ignore public and cross-account access for buckets with policies that grant public access to buckets and objects.

### Manage system permissions

[Previous](#) **Next**

6. Choose **Create bucket**.

**Create bucket**

✓ Name and region    ✓ Configure options    ✓ Set permissions    4 Review

**Name and region** Edit

**Bucket name** aleguzman-bucket    **Region** US East (N. Virginia)

**Options** Edit

<b>Versioning</b>	Enabled
<b>Server access logging</b>	Disabled
<b>Tagging</b>	0 Tags
<b>Object-level logging</b>	Disabled
<b>Default encryption</b>	None
<b>CloudWatch request metrics</b>	Disabled
<b>Object lock</b>	Disabled

**Permissions** Edit

**Block all public access**  
On

- Block public access to buckets and objects granted through *new* access control lists (ACLs)  
On
- Block public access to buckets and objects granted through *any* access control lists (ACLs)  
On

Previous 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/>.

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

## Identity and Access Management (IAM)

▼ AWS Account (511167456022)

Dashboard

Groups

Users

**Roles**

Policies

Identity providers

Account settings

Credential report





🔍 Search IAM

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

### Create role

1 2 3 4

Select type of trusted entity

 <b>AWS service</b> EC2, Lambda and others	 <b>Another AWS account</b> Belonging to you or 3rd party	 <b>Web identity</b> Cognito or any OpenID provider	 <b>SAML 2.0 federation</b> Your corporate directory
--	---	---	--

Allows AWS services to perform actions on your behalf. [Learn more](#)

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.

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

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**.

## Create role

1 2 3 4

## ▼ Attached permissions policies

The type of role that you selected requires the following policy.

Filter policies ▼ <input type="text" value="Search"/>			Showing 3 results
Policy name ▼	Used as	Description	
▶  <a href="#">AWSIoTLogging</a>	None	Allows creation of Amazon CloudWatch Log gr...	
▶  <a href="#">AWSIoTRuleActions</a>	None	Allows access to all AWS services supported i...	
▶  <a href="#">AWSIoTTThingsRegistration</a>	None	This policy allows users to register things at bu...	

## ▶ Set permissions boundary

\* Required

Cancel

Previous

Next: Tags



9. Choose **Next: Review**.

Create role



Add tags (optional)

IAM tags are key-value pairs you can add to your role. Tags can include user information, such as an email address, or can be descriptive, such as a job title. You can use the tags to organize, track, or control access for this role. [Learn more](#)

Key	Value (optional)	Remove
<input type="text" value="Add new key"/>	<input type="text"/>	

You can add 50 more tags.

Cancel Previous **Next: Review**



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

## Create role

1 2 3 4

### Review

Provide the required information below and review this role before you create it.

Role name\* OTARole

Use alphanumeric and '+=, @-\_' characters. Maximum 64 characters.

Role description Allows IoT to call AWS services on your behalf.



Maximum 1000 characters. Use alphanumeric and '+=, @-\_' characters.

Trusted entities AWS service: iot.amazonaws.com

Policies AWSIoTLogging AWSIoTRuleActions AWSIoTThingsRegistration

Permissions boundary Permissions boundary is not set

No tags were added.

\* Required

Cancel

Previous

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.

The screenshot shows the AWS Identity and Access Management (IAM) console. On the left, the 'Roles' link in the navigation menu is highlighted with a red box. The main area displays a search bar with 'OTAR' entered. Below the search bar, a table lists the roles found. The first row is 'OTARole', which is highlighted with a red arrow. The table has columns for 'Role name', 'Description', and 'Trusted entities'. The description for 'OTARole' is 'Allows IoT to call AWS services on your behalf.' and the trusted entity is 'AWS service: iot'.

Role name	Description	Trusted entities
OTARole	Allows IoT to call AWS services on your behalf.	AWS service: iot

## 2. Choose **Attach** policies.

Roles > OTARole

Summary Delete role

Role ARN: `arn:aws:iam::[redacted]:role/OTARole`

Role description: Allows IoT to call AWS services on your behalf. [Edit](#)

Instance Profile ARNs: [+](#)

Path: /

Creation time: 2019-11-05 13:04 CST

Maximum CLI/API session duration: 1 hour [Edit](#)

Permissions | Trust relationships | Tags | Access Advisor | Revoke sessions

▼ Permissions policies (3 policies applied)

**Attach policies** ➕ Add inline policy

Policy name	Policy type	
<a href="#">AWSIoTThingsRegistration</a>	AWS managed policy	✕
<a href="#">AWSIoTLogging</a>	AWS managed policy	✕

[Show 1 more](#)

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.

Add permissions to OTARole

Attach Permissions

Create policy ↺

Filter policies  Showing 1 result

Policy name	Type	Used as
<input checked="" type="checkbox"/> <a href="#">AmazonFreeRTOSOTAUpdate</a>	AWS managed	None

Cancel **Attach policy**

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

1. Choose **Add inline policy**.

Roles > OTARole

### Summary

Policy AmazonFreeRTOSOTAUpdate has been attached for the OTARole.

**Role ARN** `arn:aws:iam::<account_id>:role/OTARole`

**Role description** Allows IoT to call AWS services on your behalf. | [Edit](#)

**Instance Profile ARNs** [+](#)

**Path** /

**Creation time** 2019-11-05 13:04 CST

**Maximum CLI/API session duration** 1 hour [Edit](#)

**Permissions** | Trust relationships | Tags | Access Advisor | Revoke sessions

▼ Permissions policies (4 policies applied)

[Attach policies](#) [Add inline policy](#)

Policy name	Policy type	
<a href="#">AWSIoTThingsRegistration</a>	AWS managed policy	✕
<a href="#">AWSIoTLogging</a>	AWS managed policy	✕
<a href="#">AmazonFreeRTOSOTAUpdate</a>	AWS managed policy	✕
<a href="#">AWSIoTRuleActions</a>	AWS managed policy	✕
▶ Permissions boundary (not set)		

- Choose the **JSON** tab.
- 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**.

**Account Settings**

**Account Id:** 000000000000

**Seller:** AWS Inc.

**Account Name:** 123456789012

**Password:** \*\*\*\*\*

- Choose **Review policy**.

Create policy

12

A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the visual editor and using JSON. [Learn more](#)

Visual editor

**JSON**

Import managed policy

```

1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Action": [
7         "iam:GetRole",
8         "iam:PassRole"
9       ],
10      "Resource": "arn:aws:iam::< >:role/OTARole>"
11    }
12  ]
13 }

```

Cancel

**Review policy**

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

Create policy

12

Review policy

Before you create this policy, provide the required information and review this policy.

Name\*

OTARolePolicy

Maximum 128 characters. Use alphanumeric and '+-=, @ \_ .' characters.

Summary

Filter

Service	Access level	Resource	Request condition
Allow (1 of 203 services) <a>Show remaining 202</a>			
IAM	Limited: Read, Write	RoleName   string like   OTARole>	None

\* Required

Cancel

Previous

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**.

Roles > OTARole

Summary

Delete role

Role ARN

arn:aws:iam::< >:role/OTARole

Role description

Allows IoT to call AWS services on your behalf. | Edit

Instance Profile ARNs

Path

/

Creation time

2019-11-05 13:04 CST

Maximum CLI/API session duration

1 hour Edit

Permissions

Trust relationships

Tags

Access Advisor

Revoke sessions

Permissions policies (5 policies applied)

Attach policies

Add inline policy

Policy name	Policy type	
<a>AWSIoTThingsRegistration</a>	AWS managed policy	✕
<a>AWSIoTLogging</a>	AWS managed policy	✕
<a>AmazonFreeRTOSOTAUpdate</a>	AWS managed policy	✕
<a>AWSIoTRuleActions</a>	AWS managed policy	✕
OTARolePolicy	Inline policy	✕

Permissions boundary (not set)

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**.

## Create policy

1

2

A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the visual editor and using JSON. [Learn more](#)

Visual editor

JSON

[Import managed policy](#)

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Action": [
7         "s3:ListBucketVersions",
8         "s3:GetObjectVersion",
9         "s3:GetObject",
10        "s3:PutObject"
11      ],
12      "Resource": [
13        "arn:aws:s3::: ",
14        "arn:aws:s3::: "
15      ]
16    }
17  ]
18 }
```

Cancel

Review policy



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

### Create policy

1

2

### Review policy

Before you create this policy, provide the required information and review this policy.

Name\*

Maximum 128 characters. Use alphanumeric and '\*+=, @-\_' characters.

#### Summary

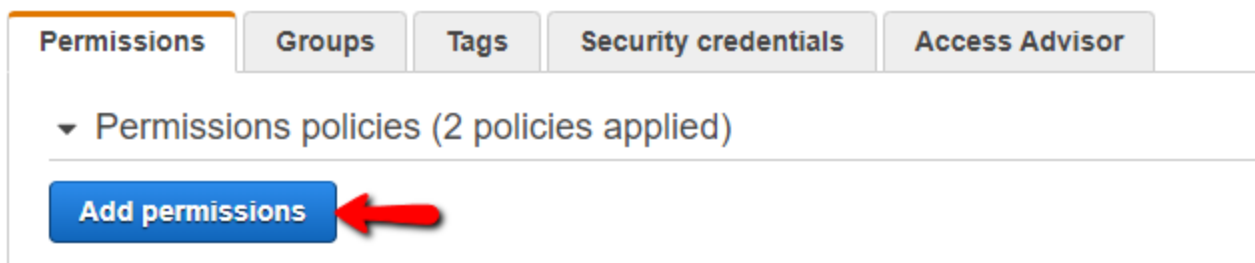
Service	Access level	Resource	Request condition
Allow (1 of 203 services) <a href="#">Show remaining 202</a>			
S3	Limited: Read, Write	Multiple	None

\* Required

[Cancel](#)[Previous](#)[Create policy](#)

## 2.3 Create an OTA User Policy

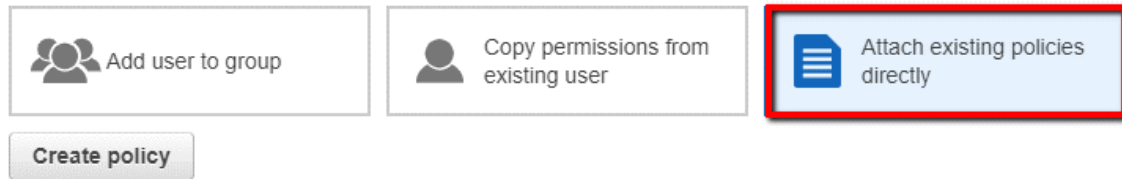
1. Open the <https://console.aws.amazon.com/iam/> console.
2. In the navigation pane, choose **Users**.
3. Choose your IAM user from the list.
4. Choose **Add permissions**.



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

## Grant permissions

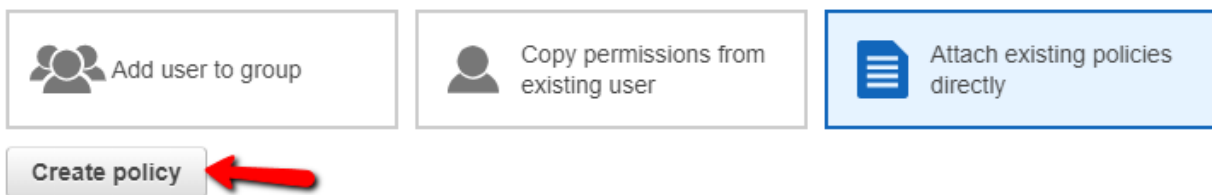
Use IAM policies to grant permissions. You can assign an existing policy or create a new one.



6. Choose **Create policy**.

## Grant permissions

Use IAM policies to grant permissions. You can assign an existing policy or create a new one.



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>/*"
    }
  ]
}
```

```

    "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**.

#### Create policy

1

2

A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the visual editor and using JSON. [Learn more](#)

Visual editor

JSON

[Import managed policy](#)

```

8      "s3:ListAllMyBuckets",
9      "s3:CreateBucket",
10     "s3:PutBucketVersioning",
11     "s3:GetBucketLocation",
12     "s3:GetObjectVersion",
13     "acm:ImportCertificate",
14     "acm:ListCertificates",
15     "iot:*",
16     "iam:ListRoles",
17     "freertos:ListHardwarePlatforms",
18     "freertos:DescribeHardwarePlatform"
19   ],
20   "Resource": "*"
21 },
22 {
23   "Effect": "Allow",
24   "Action": [
25     "s3:GetObject",
26     "s3:PutObject"
27   ],
28   "Resource": "arn:aws:s3:::<bucket>/*"
29 },
30 {
31   "Effect": "Allow",
32   "Action": "iam:PassRole",
33   "Resource": "arn:aws:iam::<account-id>:role/OTARole"
34 }
35 ]
36 }
37

```

Cancel

Review policy

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

### Create policy

1

2

#### Review policy

Name\*

Use alphanumeric and '+=, @-\_' characters. Maximum 128 characters.

Description

Maximum 1000 characters. Use alphanumeric and '+=, @-\_' characters.

#### Summary

Filter			
Service	Access level	Resource	Request condition
Allow (5 of 203 services) <a href="#">Show remaining 198</a>			
<a href="#">Certificate Manager</a>	Full: List Limited: Write	All resources	None
<a href="#">FreeRTOS</a>	Limited: List, Read	All resources	None
<a href="#">IAM</a>	Limited: List, Write	Multiple	None
<a href="#">IoT</a>	Full access	All resources	None
<a href="#">S3</a>	Limited: List, Read, Write	Multiple	None

\* Required

[Cancel](#)

[Previous](#)

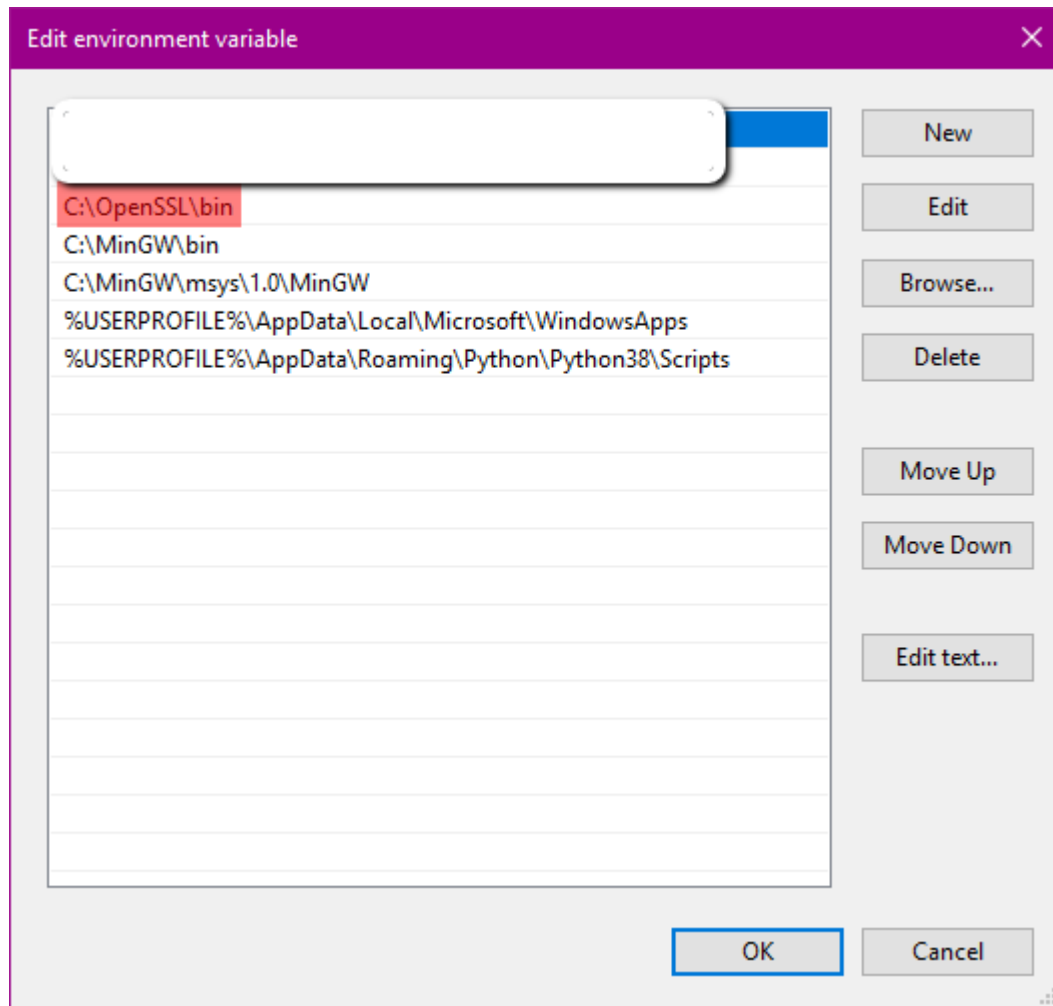
[Create policy](#)

## 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

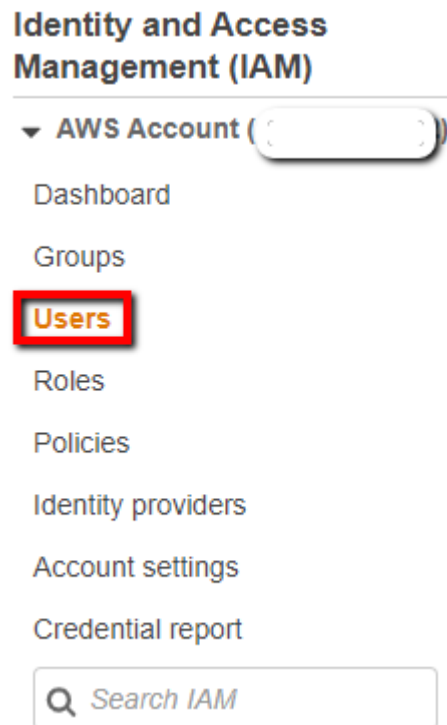


*Make sure openssl gets assigned to the OpenSSL executable in your command prompt or terminal environment*

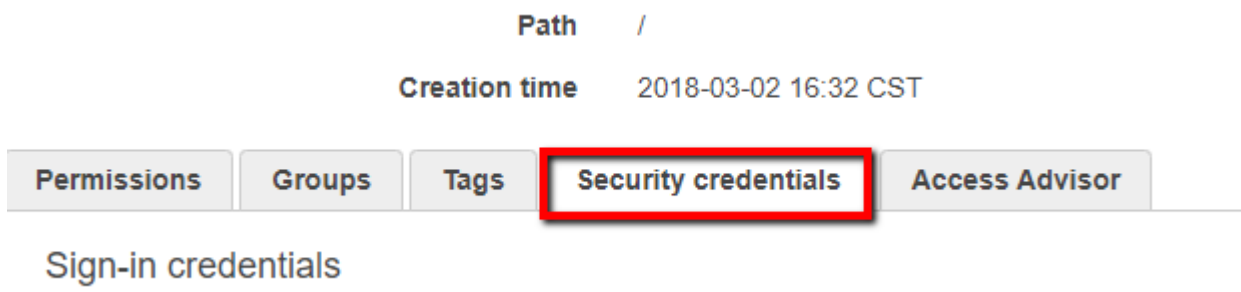
### 2.4.2 Install the AWS CLI

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/>

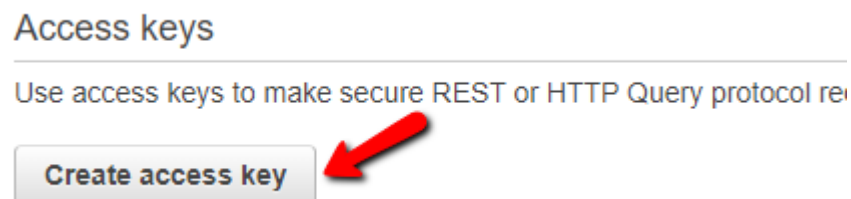
3. In the navigation pane, choose **Users**.



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



6. In the **Access 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.

Create access key ✕

✓

**Success**

This is the **only** time that the secret access keys can be viewed or downloaded. You cannot recover them later. However, you can create new access keys at any time.

↓
Download .csv file

Access key ID	Secret access key
<input style="width: 100%;" type="text"/>	<div style="display: flex; align-items: center;"> <span>*****</span> <span style="margin-left: 5px; color: #4a7ebb; font-size: 12px;">Show</span> </div>

Close

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

```

C:\>aws configure
AWS Access Key ID [None]: 
AWS Secret Access Key [None]: 
Default region name [None]: us-east-1
Default output format [None]: json
C:\>
  
```

## 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

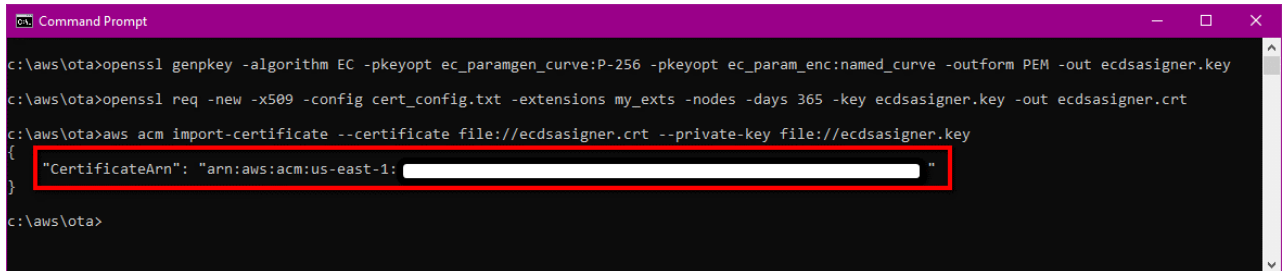
[ my\_exts ]

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.



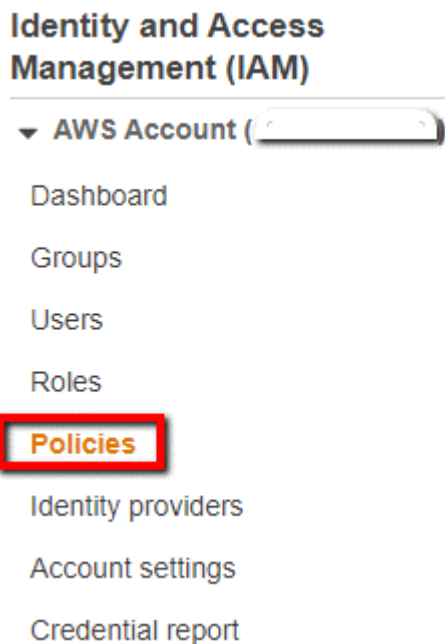
```
Command Prompt
c:\aws\ota>openssl genkey -algorithm EC -pkeyopt ec_paramgen_curve:P-256 -pkeyopt ec_param_enc:named_curve -outform PEM -out ecdsasigner.key
c:\aws\ota>openssl req -new -x509 -config cert_config.txt -extensions my_exts -nodes -days 365 -key ecdsasigner.key -out ecdsasigner.crt
c:\aws\ota>aws acm import-certificate --certificate file://ecdsasigner.crt --private-key file://ecdsasigner.key
{
  "CertificateArn": "arn:aws:acm:us-east-1: [redacted]"
}
c:\aws\ota>
```



## 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**.

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**.

## Create policy

1

2

A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the visual editor and using JSON. [Learn more](#)

Visual editor

JSON

[Import managed policy](#)

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Effect": "Allow",  
6       "Action": [  
7         "signer:*"  
8       ],  
9       "Resource": "*"   
10    }  
11  ]  
12 }  
13
```

Cancel

Review policy



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

## Create policy

1

2

### Review policy

Name\*

Use alphanumeric and '+', '@', '-' characters. Maximum 128 characters.

Description

Maximum 1000 characters. Use alphanumeric and '+', '@', '-' characters.

### Summary

Service ▼

Access level

Resource

Request condition

Allow (1 of 203 services) [Show remaining 202](#)

[Signer](#)

Full access

All resources

None

\* Required

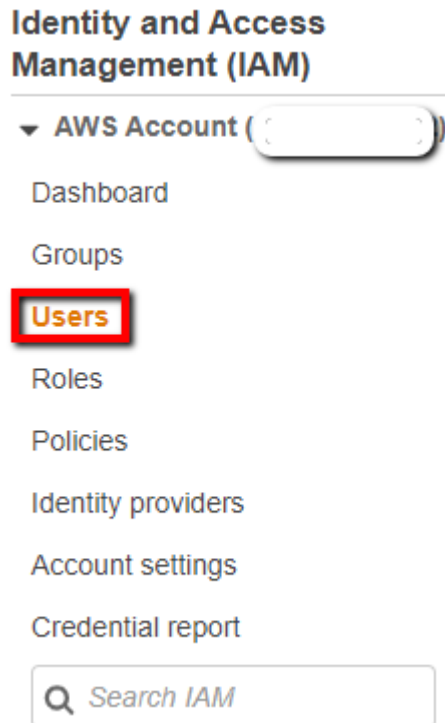
[Cancel](#)

[Previous](#)

[Create policy](#)



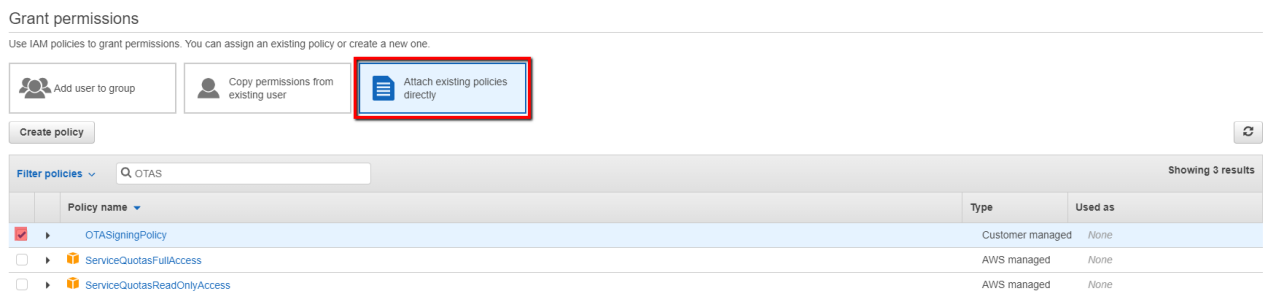
3. In the navigation pane, choose **Users**.



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

**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**.

**Next: Review**

8. Choose **Add permissions**.

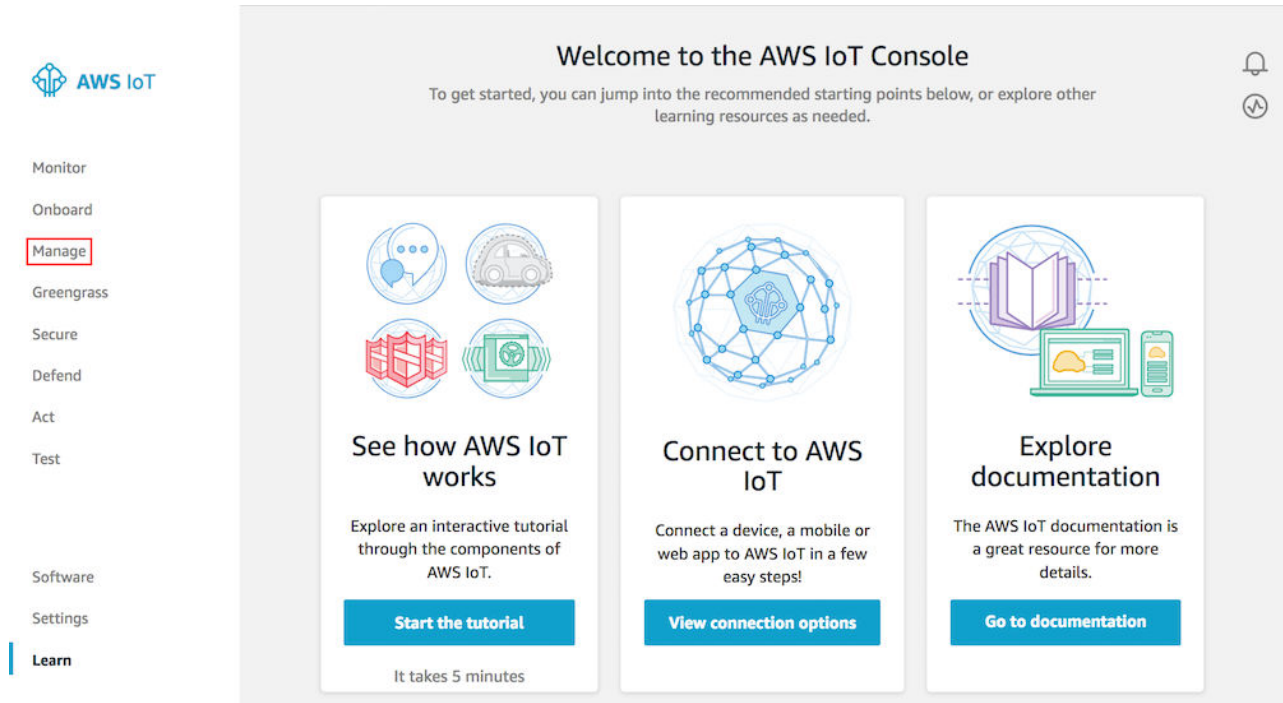
**Add permissions**

# 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**.



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



## You don't have any things yet

A thing is the representation of a device in the cloud.

[Learn more](#)
[Register a thing](#)

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

## Creating AWS IoT things

An IoT thing is a representation and record of your physical device in the cloud. Any physical device needs a thing record in order to work with AWS IoT. [Learn more](#).

### Register a single AWS IoT thing

Create a thing in your registry

[Create a single thing](#)

### Bulk register many AWS IoT things

Create things in your registry for a large number of devices already using AWS IoT, or register devices so they are ready to connect to AWS IoT.

[Create many things](#)
[Cancel](#)
[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**.

CREATE A THING

STEP 1/3

Add your device to the thing registry

This step creates an entry in the thing registry and a thing shadow for your device.

Name

myThing

---

**Apply a type to this thing**

Using a thing type simplifies device management by providing consistent registry data for things that share a type. Types provide things with a common set of attributes, which describe the identity and capabilities of your device, and a description.

Thing Type

No type selected

Create a type

---

**Add this thing to a group**

Adding your thing to a group allows you to manage devices remotely using jobs.

Thing Group

Groups /

Create group Change

---

**Set searchable thing attributes (optional)**

Enter a value for one or more of these attributes so that you can search for your things in the registry.

Attribute key	Value	
<div>Provide an attribute key, e.g. Manufacturer</div>	<div>Provide an attribute value, e.g. Acme-Corporation</div>	<div>Clear</div>
<div>Add another</div>		

Show thing shadow ▾

Cancel

Back

Next

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

CREATE A THING

## Add a certificate for your thing

STEP 2/3

A certificate is used to authenticate your device's connection to AWS IoT.

**One-click certificate creation (recommended)**  
This will generate a certificate, public key, and private key using AWS IoT's certificate authority.

**Create certificate**

---

**Create with CSR**  
Upload your own certificate signing request (CSR) based on a private key you own.

**Create with CSR**

---

**Use my certificate**  
Register your CA certificate and use your own certificates for one or many devices.

**Get started**

---

**Skip certificate and create thing**  
You will need to add a certificate to your thing later before your device can connect to AWS IoT.

**Create thing without certificate**

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**.

## Certificate created!

Download these files and save them in a safe place. Certificates can be retrieved at any time, but the private and public keys cannot be retrieved after you close this page.

In order to connect a device, you need to download the following:

A certificate for this thing	c3c4ff2375.cert.pem	<a href="#">Download</a>
A public key	c3c4ff2375.public.key	<a href="#">Download</a>
A private key	c3c4ff2375.private.key	<a href="#">Download</a>

You also need to download a root CA for AWS IoT:

A root CA for AWS IoT [Download](#)

[Activate](#)

[Cancel](#)

[Done](#)

[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.

CREATE A THING

## Add a policy for your thing

STEP  
3/3

Select a policy to attach to this certificate:

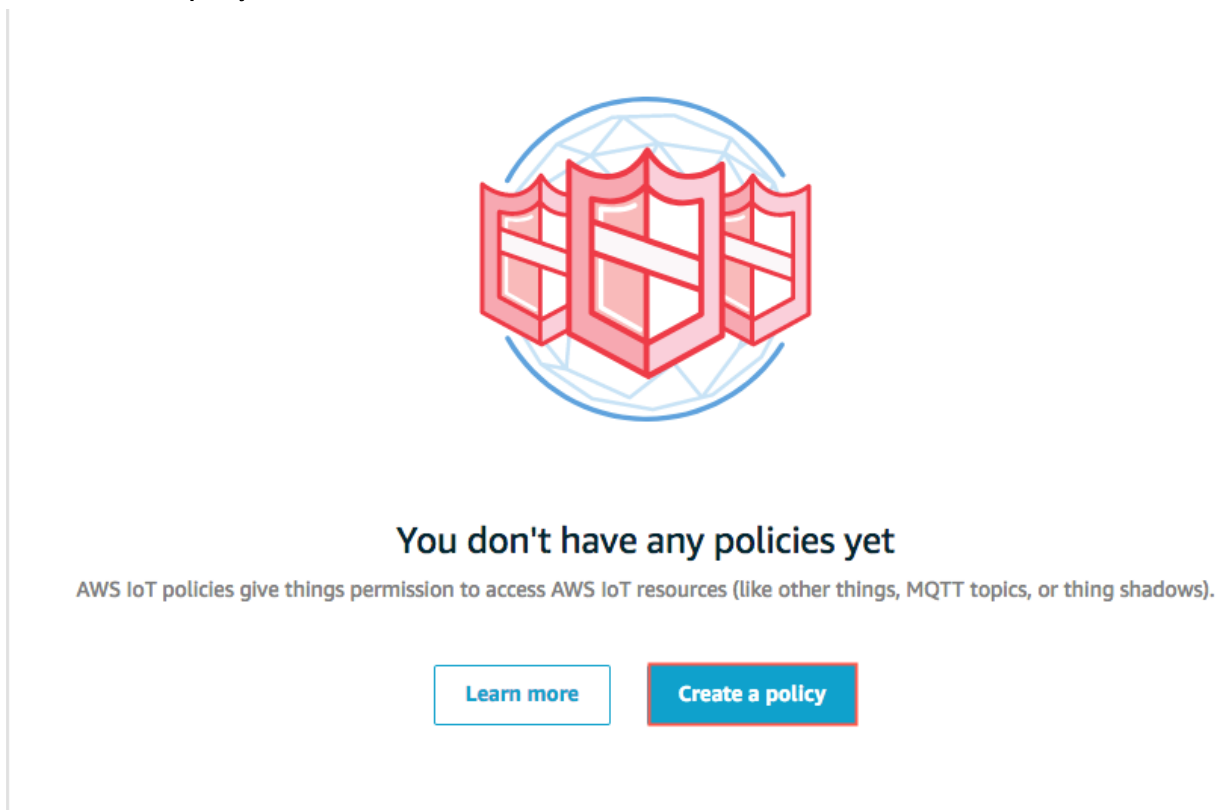
**No match found**  
There are no policies in your account.

0 policies selected

[Register Thing](#)

## 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**.

## Create a policy

Create a policy to define a set of authorized actions. You can authorize actions on one or more resources (things, topics, topic filters). To learn more about IoT policies go to the [AWS IoT Policies documentation page](#).

Name

myIoTPolicy

### Add statements

Policy statements define the types of actions that can be performed by a resource.

Advanced mode

Action

iot:\*

Resource ARN

\*

Effect



Allow



Deny

Remove

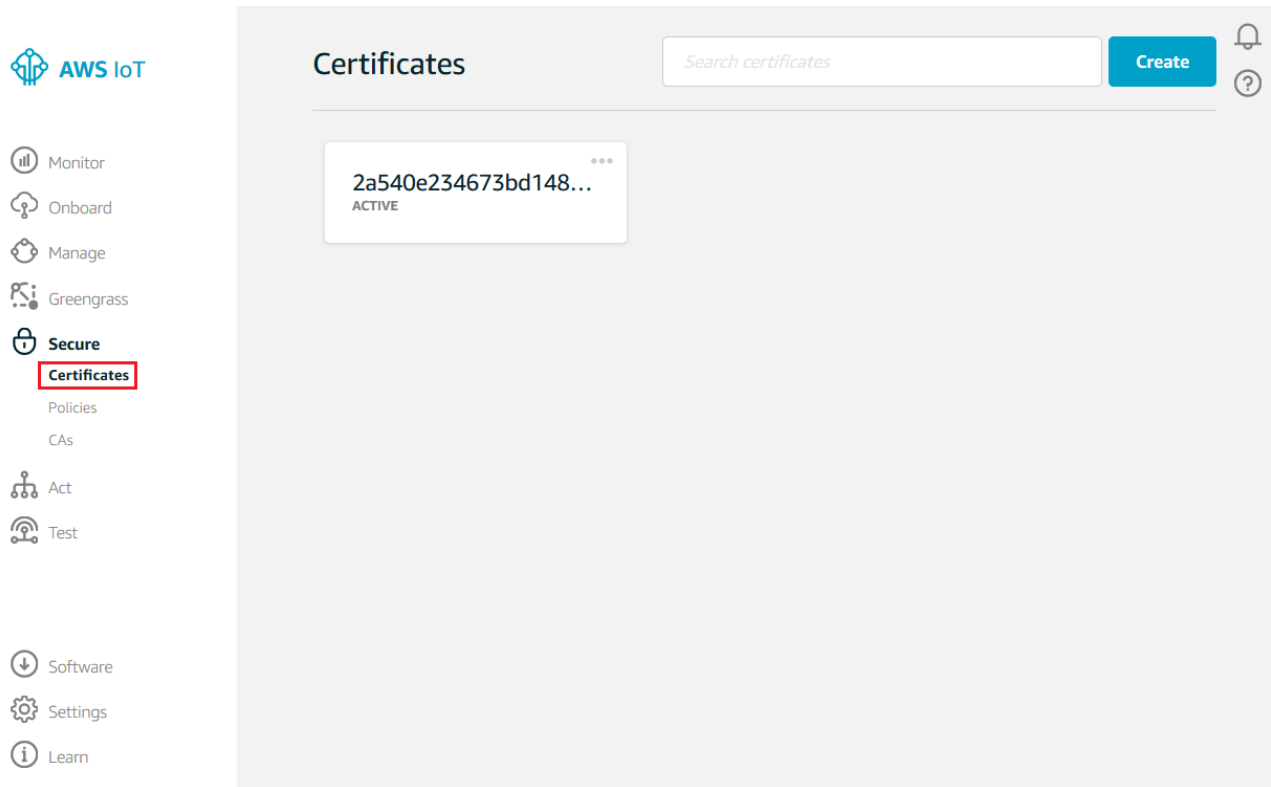
Add statement

Create

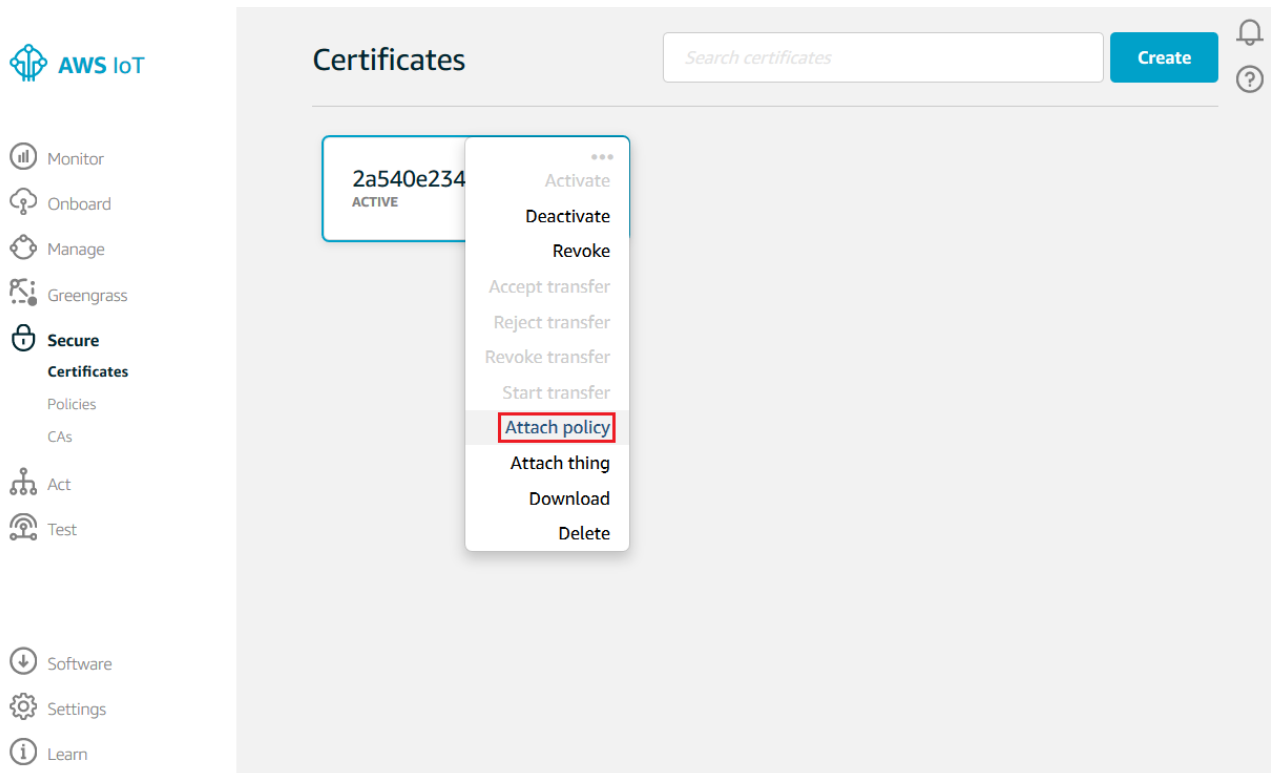


### 4.3 Attach an AWS IoT Policy to a Device Certificate

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**.

## Attach policies to certificate(s)

Policies will be attached to the following certificate(s):

Choose one or more policies

☒ myIoTPolicy [View](#)

1 policy selected [Cancel](#) [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**.

AWS IoT

Monitor

Onboard

Manage

Greengrass

**Secure**

**Certificates**

Policies

CAs

Act

Test

Software

Settings

Learn

### Certificates

Search certificates

Create

2a540e234  
ACTIVE

...

Activate

Deactivate

Revoke

Accept transfer

Reject transfer

Revoke transfer

Start transfer

Attach policy

**Attach thing**

Download

Delete

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

### Attach things to certificate(s)


Things will be attached to the following certificate(s):

Choose one or more things

☒ myThing

1 thing selected Cancel Attach

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



- Monitor
- Onboard
- Manage
- Greengrass
- Secure**
  - Certificates**
  - Policies
  - CAs
- Act
- Test
- Software
- Settings
- Learn

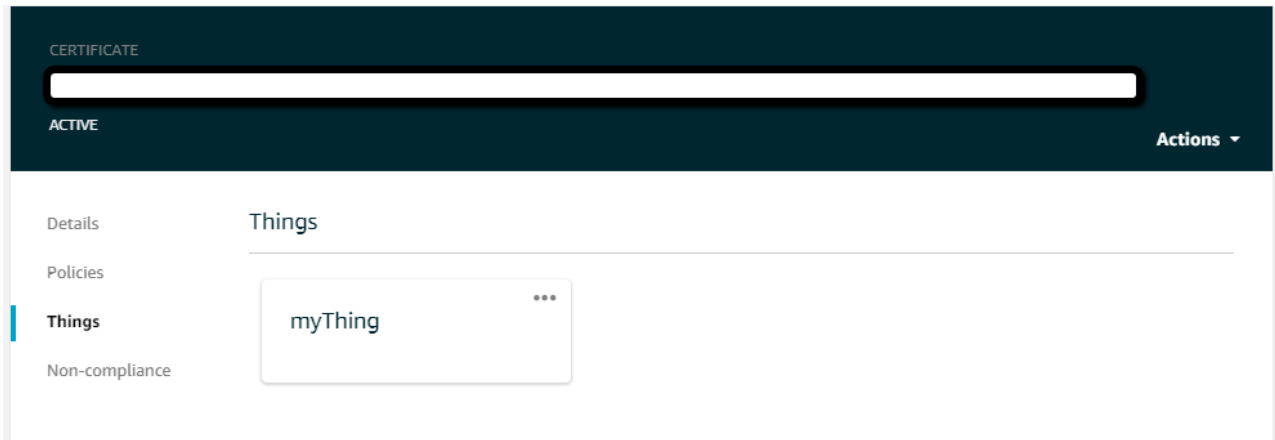
### Certificates

Create

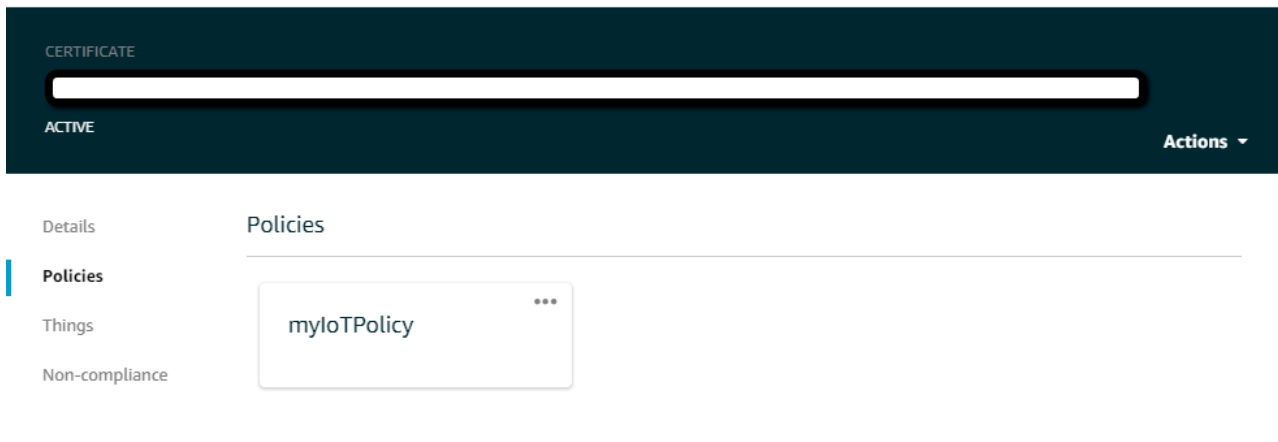
2a540e234673bd148...

ACTIVE

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**.



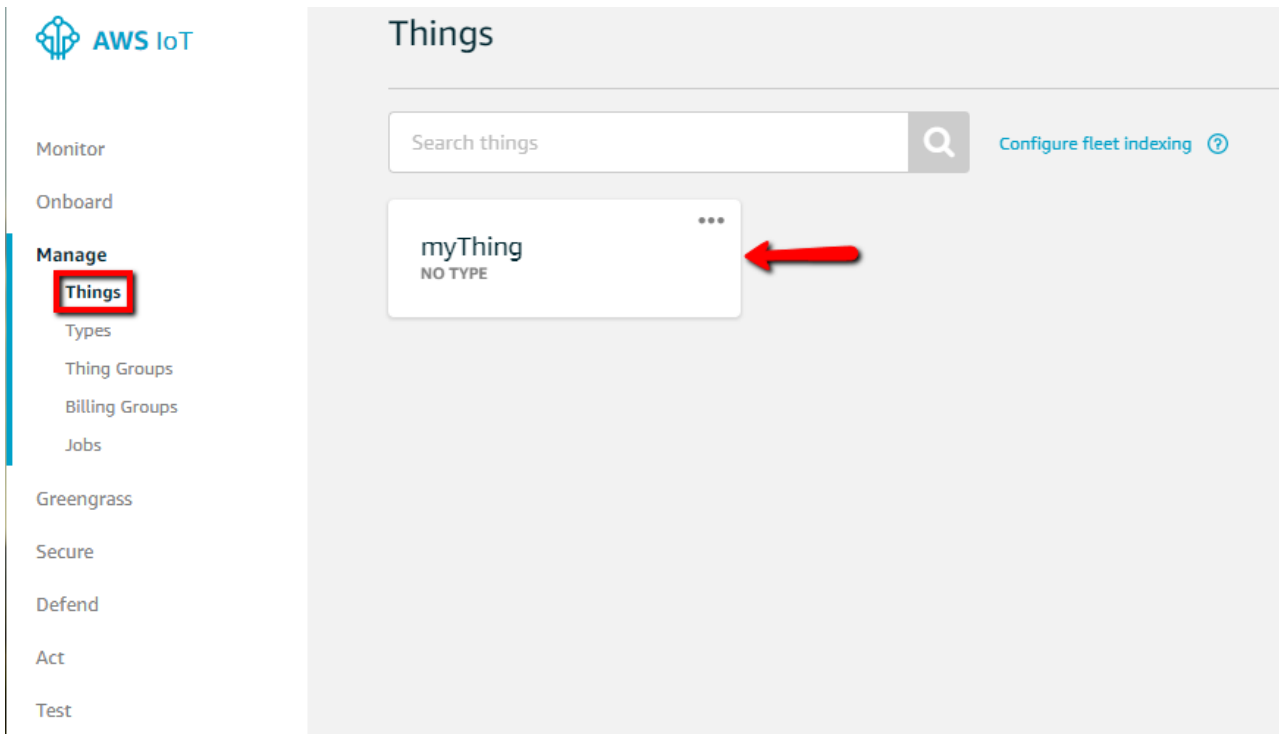
## Chapter 5

# 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

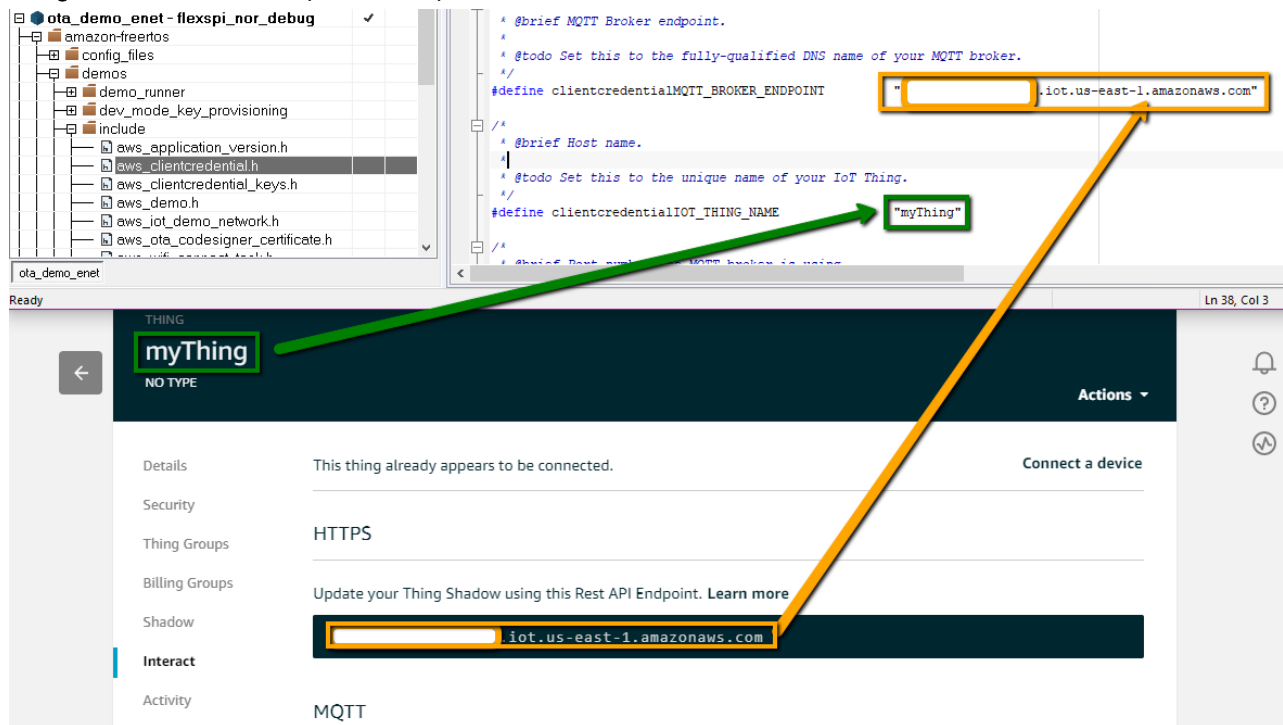
1. Open the AWS IoT console website <https://console.aws.amazon.com/iot/>
2. On the **Welcome to the AWS IoT Console** page, in the navigation pane, choose **Manage – Things** select the previously created **Thing**.



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

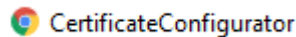


- Inside the OTA project, open `amazon-freertos - demos - include - aws_clientcredential.h` and set the **REST\_API** and **IoT Thing** name obtained in the previous step.



## 5.2 aws\_clientcredential\_keys.h

- In your PC go to `...rtos\amazon-freertos\tools\certificate_configuration`
- Using a web browser open the `CertificateConfigurator.html`
- Browse to the Certificate and Key files previously downloaded from your Thing and click on **Generate and save** `aws_clientcredential_keys.h`



## Certificate Configuration Tool

Amazon FreeRTOS Developer Demos


Provide client certificate and private key PEM files downloaded from the AWS IoT Console.


**Certificate PEM file:**

Choose File

**Private Key PEM file:**

Choose File



 Generate and save `aws_clientcredential_keys.h`

- Using windows explorer replace the...\rtos\amazon-freertos\demos\include\aws\_clientcredential\_keys.h with the file obtained in the previous step.

### 5.3 aws\_ota\_codesigner\_certificate

- Open the ecdsaigner.crt file using a text editor
- 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

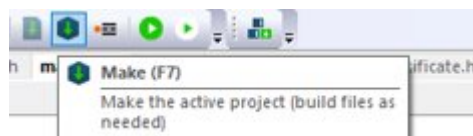
```

/*
 * 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"
"GEFsZWphbmRyYS5HdXptYW5AbnhwLmNvbTAeFw0xOTEwMjMxNDJaFw0yMDEw\n"
"MjIxNDJaMCMxITAfBgNVBAMMGEFsZWphbmRyYS5HdXptYW5AbnhwLmNvbTBZ\n"
"MBMGByqGSM49AgEGCCqGSM49AwEHA0IABGUghBD5lmF1J3wf4LYsQ2VgOaDpg98G\n"
"dNC38FWGS7owT4NC5848JumrD8SonnnXpu77Pt7ShuW39hC3Vdi7z1GjJDAiMaS\n"
"AlUdDwQEAwIHgDATBgNVHSUEDDAKBggrBgEFBQcDAzAKBggqhkhjOPQQDAgNIADBF\n"
"AiEArOpNzlaMax4arCPNiW9HYFdQTvUGyZdRLcDrUo1/LQoCIH2U2REoZ59V7r6z\n"
"CMLfHA+kWq84IjxDUE20gV60RVvC\n"
"-----END CERTIFICATE-----\n";

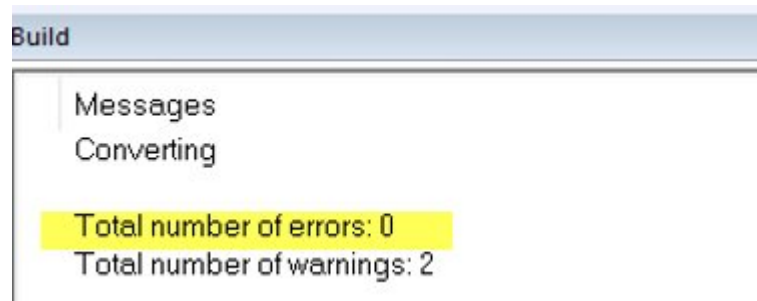
```

### 5.4 Build

- Click the make button to start building the application



- If build is successful Zero errors message is printed in build console

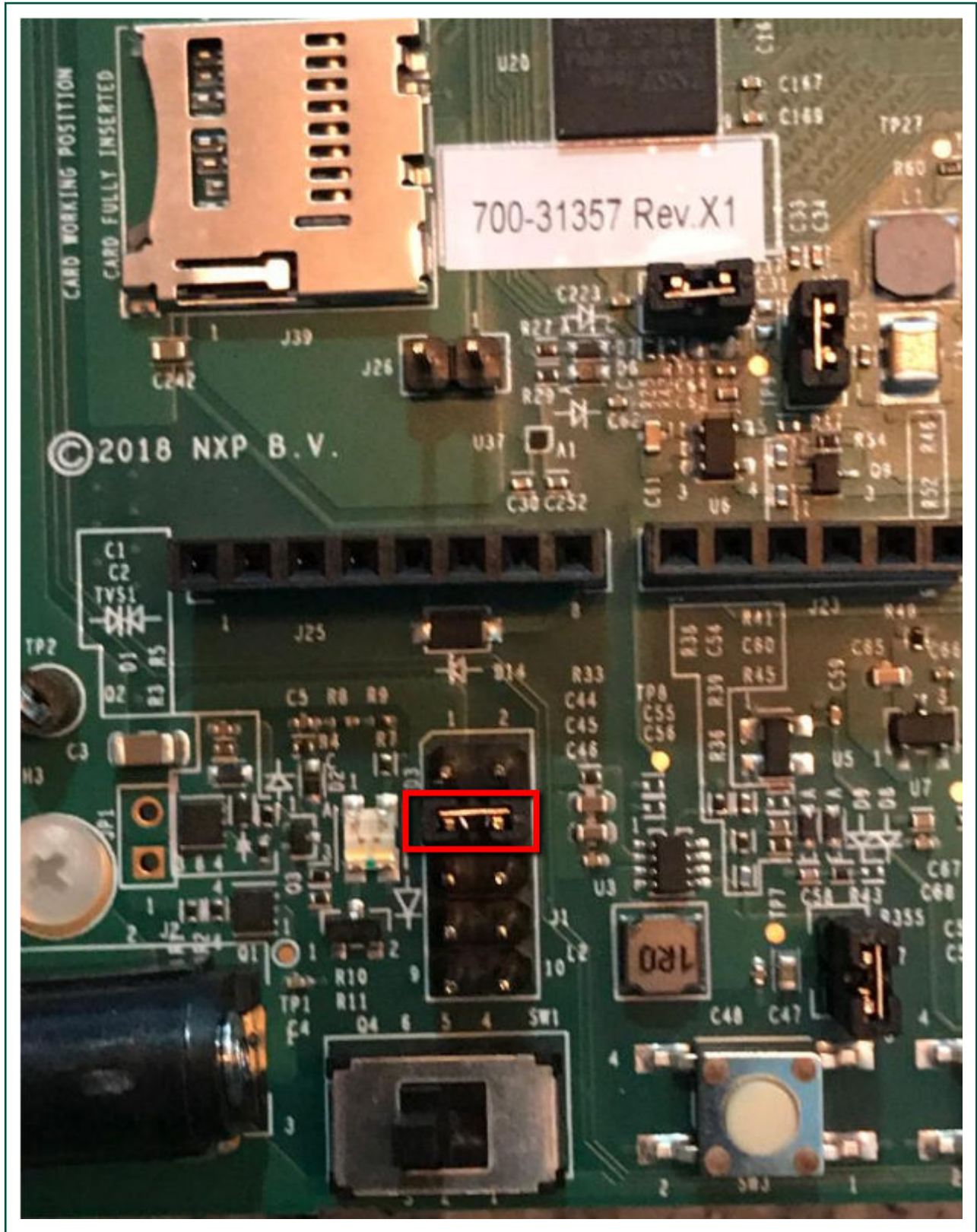


## 5.5 Programming mcu-boot into flash

1. Set all SW7 positions to off.

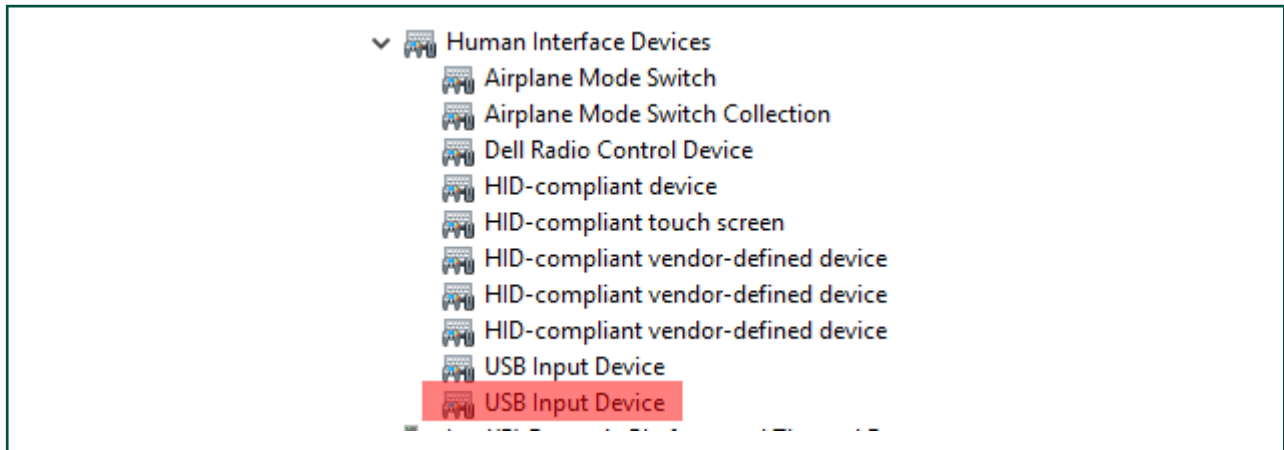


2. Locate J1, then move the jumper to **3-4**.

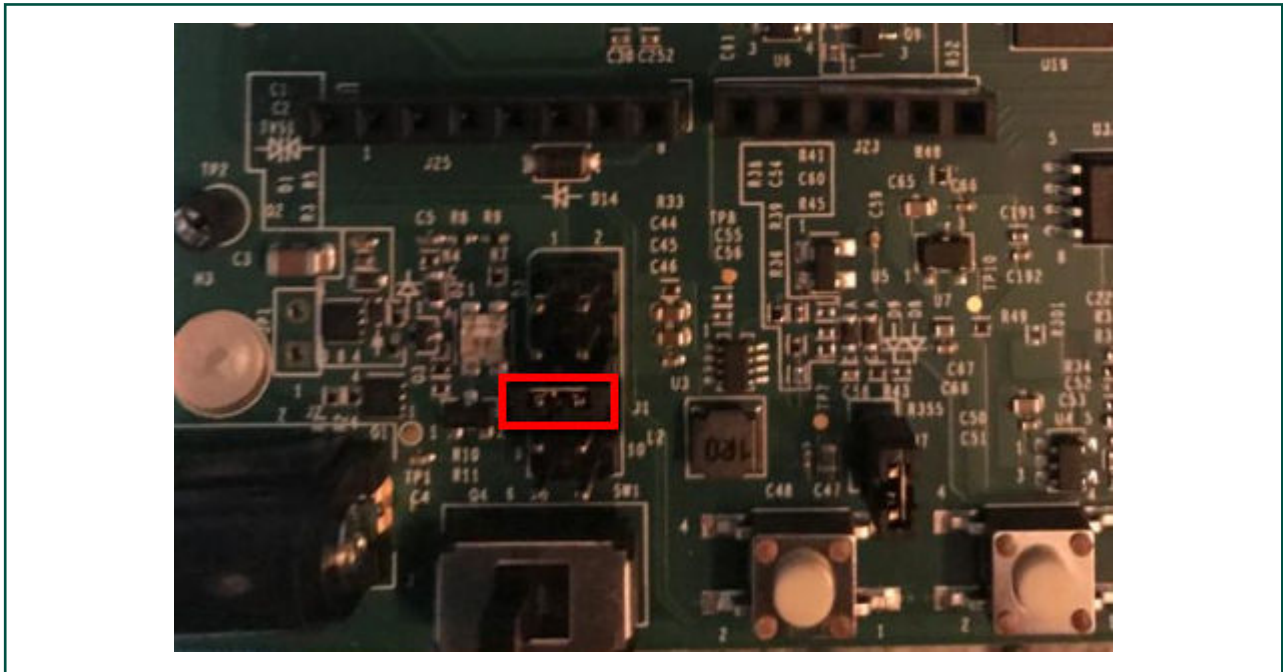


3. Connect the board to PC via J9 USB connector
4. Reset the board using **SW3**, then make sure that your RT1060-EVK gets enumerated like Human Interface Devices – USB Input device

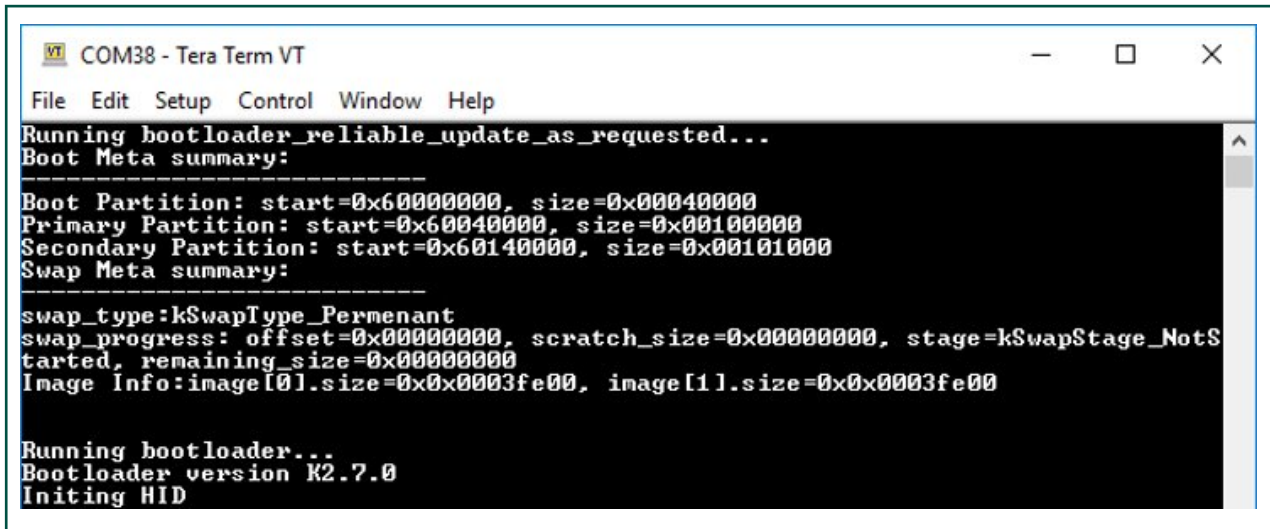




5. Open a windows Command Prompt then execute the following commands It is recommended (but not required) to have bash interpreter at hand, git bash would do the job <https://gitforwindows.org/> > cd ..\OTA\_Bootloader\_Scripts-4e081f \OTA\_Bootloader\_Scripts\_0.5 > generate\_ota\_bootloader\_and\_program\_it\_to\_flash.sh.
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



```

COM38 - Tera Term VT
File Edit Setup Control Window Help
Running bootloader_reliable_update_as_requested...
Boot Meta summary:
-----
Boot Partition: start=0x60000000, size=0x00040000
Primary Partition: start=0x60040000, size=0x00100000
Secondary Partition: start=0x60140000, size=0x00101000
Swap Meta summary:
-----
swap_type:kSwapType_Permanent
swap_progress: offset=0x00000000, scratch_size=0x00000000, stage=kSwapStage_NotS
tarted, remaining_size=0x00000000
Image Info:image[0].size=0x0x0003fe00, image[1].size=0x0x0003fe00

Running bootloader...
Bootloader version K2.7.0
Initing HID
  
```

## 5.6 Flashing the OTA Agent application

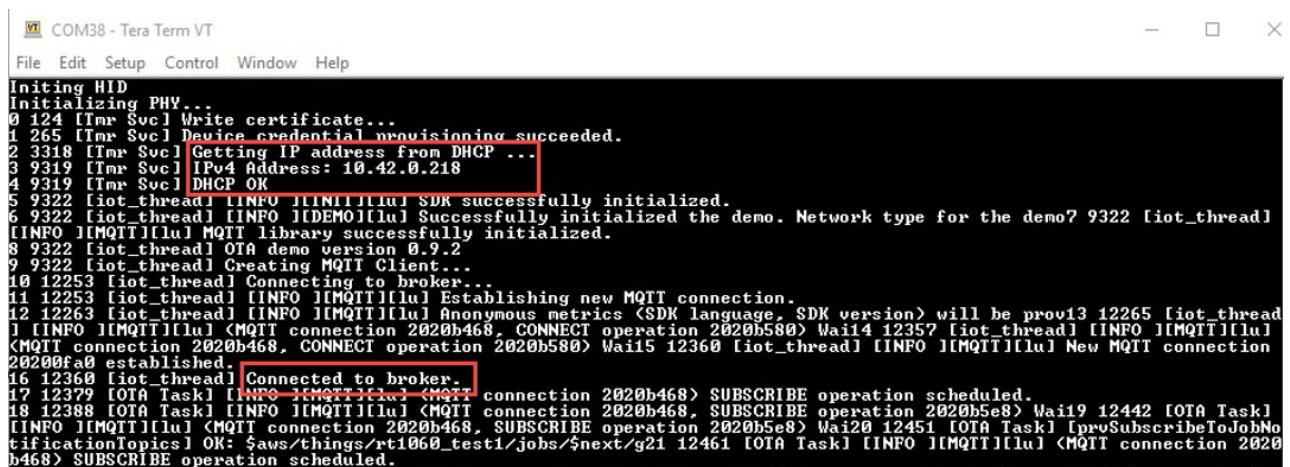
1. Attach ethernet cable with Internet connection and local DHCP server
2. Click the download and debug button to start flashing the device



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



```

COM38 - Tera Term VT
File Edit Setup Control Window Help
Initing HID
Initializing PHY...
0 124 [Tmr Svc] Write certificate...
1 265 [Tmr Svc] Device credential provisioning succeeded.
2 3318 [Tmr Svc] Getting IP address from DHCP ...
3 9319 [Tmr Svc] IPv4 Address: 10.42.0.218
4 9319 [Tmr Svc] DHCP OK
5 9322 [iot_thread] [INFO] [IMQTT] SDK successfully initialized.
6 9322 [iot_thread] [INFO] [IDEMO] Successfully initialized the demo. Network type for the demo? 9322 [iot_thread]
[INFO] [IMQTT] MQTT library successfully initialized.
8 9322 [iot_thread] OTA demo version 0.9.2
9 9322 [iot_thread] Creating MQTT Client...
10 12253 [iot_thread] Connecting to broker...
11 12253 [iot_thread] [INFO] [IMQTT] Establishing new MQTT connection.
12 12263 [iot_thread] [INFO] [IMQTT] Anonymous metrics (SDK language, SDK version) will be provi13 12265 [iot_thread]
[INFO] [IMQTT] MQTT connection 2020b468, CONNECT operation 2020b580) Wait14 12357 [iot_thread] [INFO] [IMQTT] New MQTT connection
2020fa0 established.
16 12360 [iot_thread] Connected to broker.
17 12379 [OTA Task] [INFO] [IMQTT] MQTT connection 2020b468) SUBSCRIBE operation scheduled.
18 12388 [OTA Task] [INFO] [IMQTT] MQTT connection 2020b468, SUBSCRIBE operation 2020b5e8) Wait19 12442 [OTA Task]
[INFO] [IMQTT] MQTT connection 2020b468, SUBSCRIBE operation 2020b5e8) Wait20 12451 [OTA Task] [prvSubscribeToJobNo
tificationTopics] OK: $aws/things/rt1060_test1/jobs/$next/g21 12461 [OTA Task] [INFO] [IMQTT] MQTT connection 2020
b468) SUBSCRIBE operation scheduled.
  
```

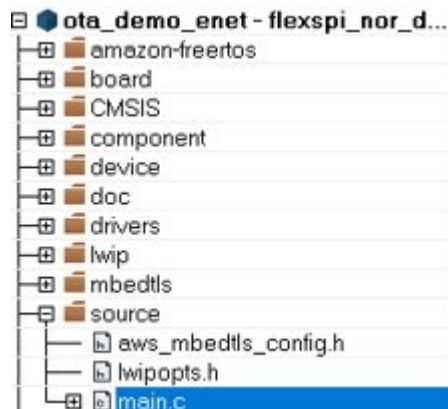
5. Stop the debug session

# Chapter 6

## OTA

### 6.1 Create new image

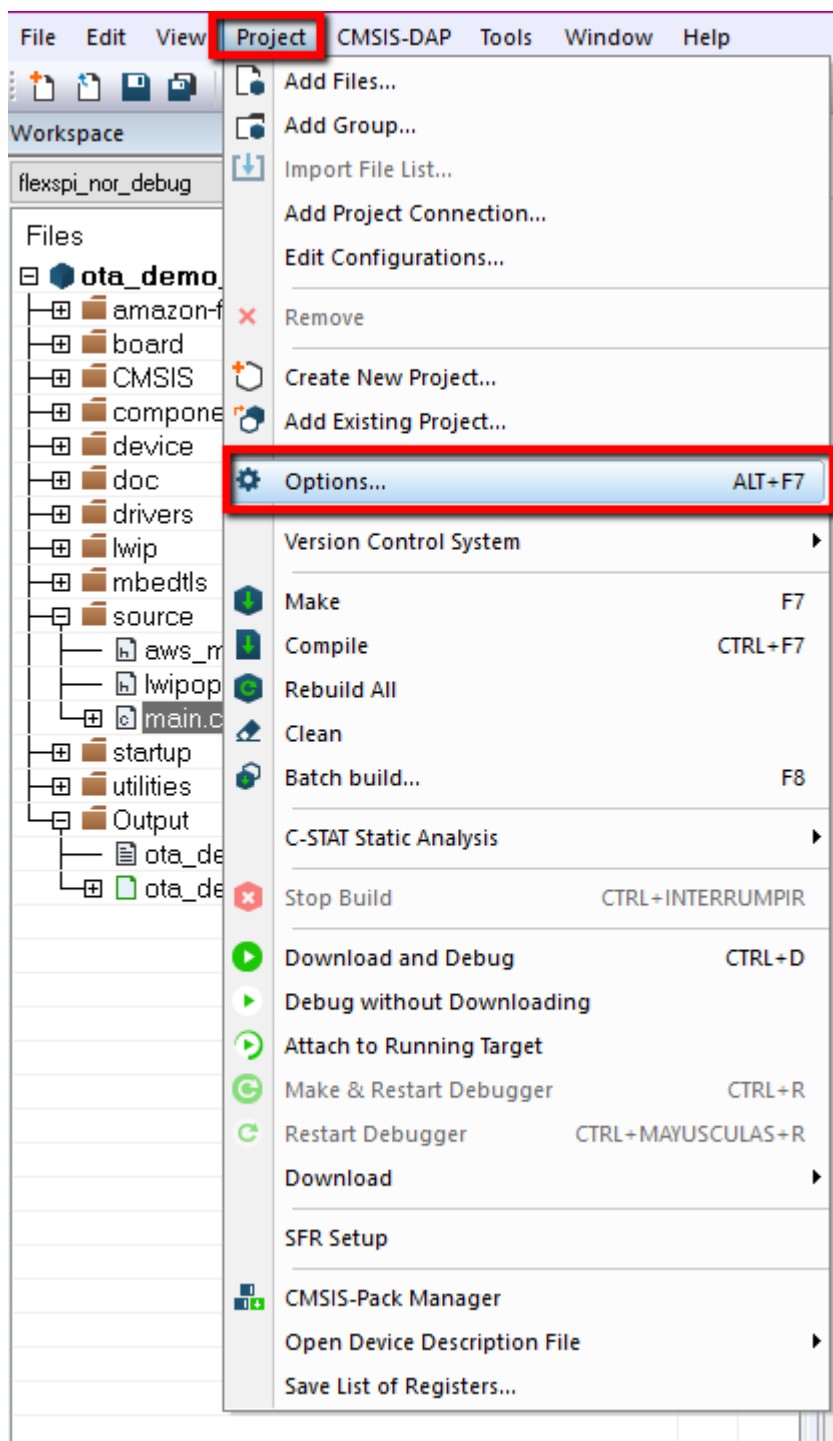
1. Open main.c



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

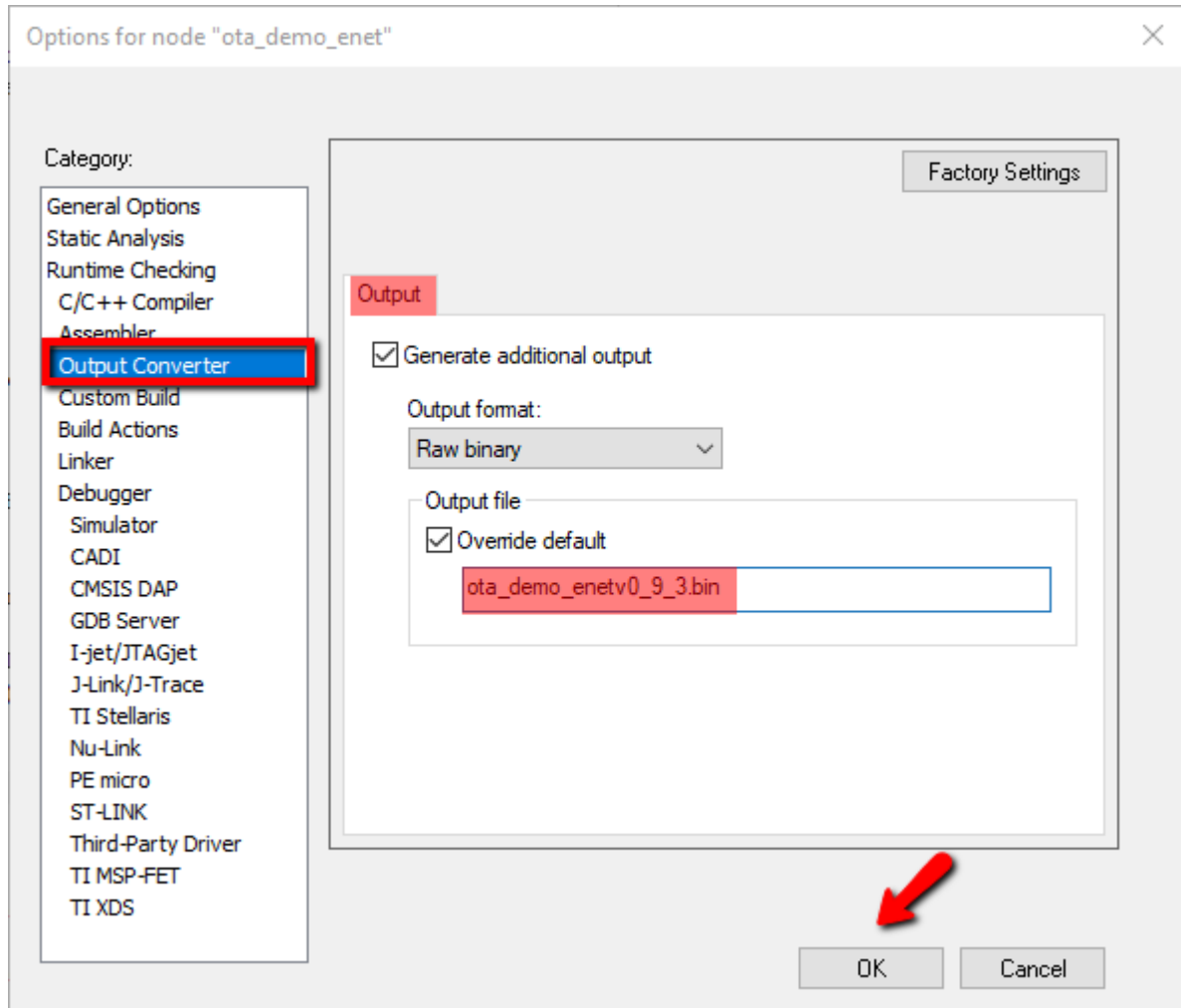
```
#define APP_VERSION_MAJOR 0
#define APP_VERSION_MINOR 9
#define APP_VERSION_BUILD 3//2
```

## 3. Open Project □ Options...

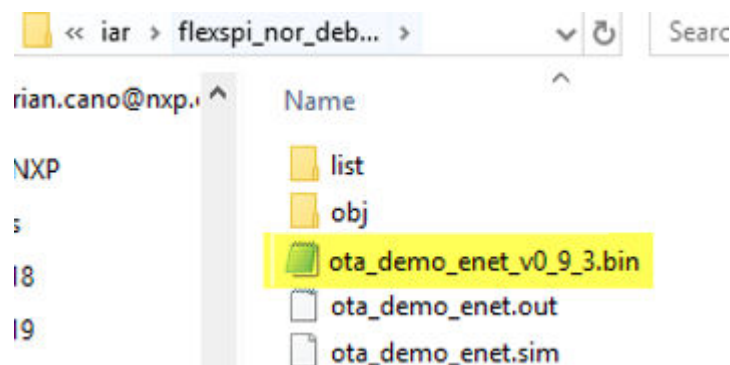




4. In the Category section choose **Output Converter** then change the name of the binary so it matches the version change then click **OK**.



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



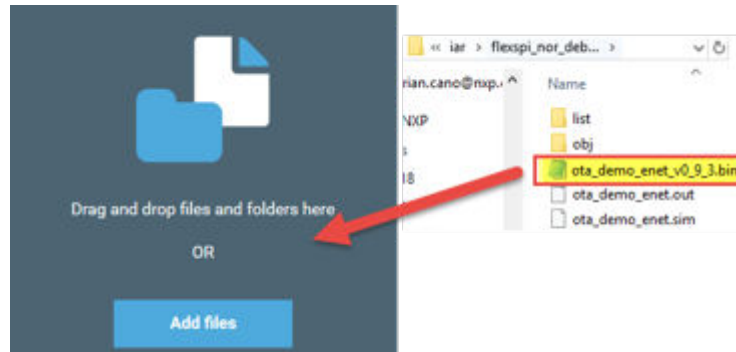
## 6.2 Uploading the binary to the S3 bucket

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](#)

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

CREATE JOB

Select a job

AWS IoT Device Management job orchestration and notification service allows you to define a set of remote operations called jobs that are sent to and executed on one or more devices connected to AWS IoT.

Create a custom job

Send a request to acquire an executable job file from one of your S3 buckets to one or more devices connected to AWS IoT.

Create custom job

Create an Amazon FreeRTOS OTA update job

This Over-the-air (OTA) update job will send your firmware image securely over MQTT to Amazon FreeRTOS-based devices

Create OTA update job

Create a Greengrass Core update job

Create a snapshot job to update one or more Greengrass Core devices with the latest Greengrass Core or OTA agent version.

Create Core update job

Cancel

Create custom job

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

CREATE JOB

## Create an Amazon FreeRTOS OTA update job

This Over-the-air (OTA) update job will send your firmware image securely over MQTT to Amazon FreeRTOS-based devices.

### Select devices to update

Browse and select the devices you want to include in this job.

No devices or thing groups selected

Close

Things	Thing Groups	Summary
<div>Q</div>		
<input type="checkbox"/>	myThing	

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

#### Select devices to update

Browse and select the devices you want to include in this job.

1 thing(s) and 0 thing group(s) selected.

Close

Things

Thing Groups

Summary

☒ myThing

Cancel

Back

Next

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

CREATE JOB

Create an Amazon FreeRTOS OTA update job

Select and sign your firmware image

Code signing ensures that devices only run code published by trusted authors and that the code has not been altered or corrupted since it was signed. You have three options for code signing. [Learn more](#)

☒ Sign a new firmware image for me


☐ Select a previously signed firmware image

☐ Use my custom signed firmware image

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

Code signing profile [Learn more](#)

No code signing profile selected

 [Create](#) [Select](#)

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

## Create a code signing profile

Profile name

myOTACodeSigning

Device hardware platform

Windows Simulator

SHA256

ECDSA

[Change](#)

Code signing certificate

AWS Certificate Manager (ACM) handles the complexity of creating and managing or importing SSL/TLS certificates. You use ACM to create an ACM Certificate or import a third-party certificate that you use for signing. You must have a certificate to sign code.

No certificate selected

[Import](#) [Select](#)

Pathname of code signing certificate on device

This is the platform-specific location and name of the certificate used by the Amazon FreeRTOS device firmware to perform OTA image signature verification.

e.g. /certificates/authcert.pem

[Cancel](#)

[Create](#)

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

## Create a code signing profile

### Profile name

myOTACodeSigning

### Device hardware platform

Windows Simulator

SHA256

ECDSA

[Change](#)

### Code signing certificate

AWS Certificate Manager (ACM) handles the complexity of creating and managing or importing SSL/TLS certificates. You use ACM to create an ACM Certificate or import a third-party certificate that you use for signing. You must have a certificate to sign code.

No certificate selected

[Close](#)

#### Select Certificate

Choose File

ecdsasigner.crt

#### Select Certificate private key

Choose File

ecdsasigner.key

#### Select Certificate chain (optional)

Choose File

No file chosen

[Import](#)





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

## Create a code signing profile

### Profile name

### Device hardware platform

[Change](#)

### Code signing certificate

AWS Certificate Manager (ACM) handles the complexity of creating and managing or importing SSL/TLS certificates. You use ACM to create an ACM Certificate or import a third-party certificate that you use for signing. You must have a certificate to sign code.

[Clear](#) [Change](#)

### Pathname of code signing certificate on device

This is the platform-specific location and name of the certificate used by the Amazon FreeRTOS device firmware to perform OTA image signature verification.

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

Select your firmware image in S3 or upload it

[Change](#)

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

CREATE JOB

## Create an Amazon FreeRTOS OTA update job

### Select and sign your firmware image

Code signing ensures that devices only run code published by trusted authors and that the code has not been altered or corrupted since it was signed. You have three options for code signing. [Learn more](#)

- ☒ Sign a new firmware image for me
- ☐ Select a previously signed firmware image
- ☐ Use my custom signed firmware image

Code signing profile [Learn more](#)

myOTACodeSigning	SHA256	ECDSA	/certificates/authcert.pem	Clear	Change
------------------	--------	-------	----------------------------	-------	--------

Select your firmware image in S3 or upload it

ota_demo_enetv0_9_3.bin	Change
-------------------------	--------

Pathname of firmware image on device [Learn more](#)

/devices/updates

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

### IAM role for OTA update job

Choose a role which grants AWS IoT access to the S3, AWS IoT jobs and AWS Code signing resources to create an OTA update job. [Learn more](#)

Role (requires S3 access)

OTARole	Select
---------	--------

13. Choose **Next**.

CREATE JOB

Create an Amazon FreeRTOS OTA update job

Select and sign your firmware image

Code signing ensures that devices only run code published by trusted authors and that the code has not been altered or corrupted since it was signed. You have three options for code signing. [Learn more](#)

☒ Sign a new firmware image for me

☐ Select a previously signed firmware image

☐ Use my custom signed firmware image

Code signing profile

[Learn more](#)

myOTACodeSigning

SHA256

ECDSA

/certificates/authcert.pem

Clear

Change

Select your firmware image in S3 or upload it

ota\_demo\_enetv0\_9\_3.bin

Change

Pathname of firmware image on device

[Learn more](#)

/devices/updates

IAM role for OTA update job

Choose a role which grants AWS IoT access to the S3, AWS IoT jobs and AWS Code signing resources to create an OTA update job. [Learn more](#)

Role (requires S3 access)

OTARole

Select

Cancel

Back

Next

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

Job type

A job can run on the devices and/or groups selected, or remain open, and apply to devices later added to a group.

☒ Your job will complete after deploying to the selected devices/groups (snapshot)

☐ Your job will continue deploying to any devices added to the selected groups (continuous)

## 15. Enter an ID for your OTA update job

**CREATE JOB**  
**Create an Amazon FreeRTOS OTA update job**

**Job type**  
A job can run on the devices and/or groups selected, or remain open, and apply to devices later added to a group.  
☒ Your job will complete after deploying to the selected devices/groups (snapshot)  
☐ Your job will continue deploying to any devices added to the selected groups (continuous)

**ID**

**Description (optional)**

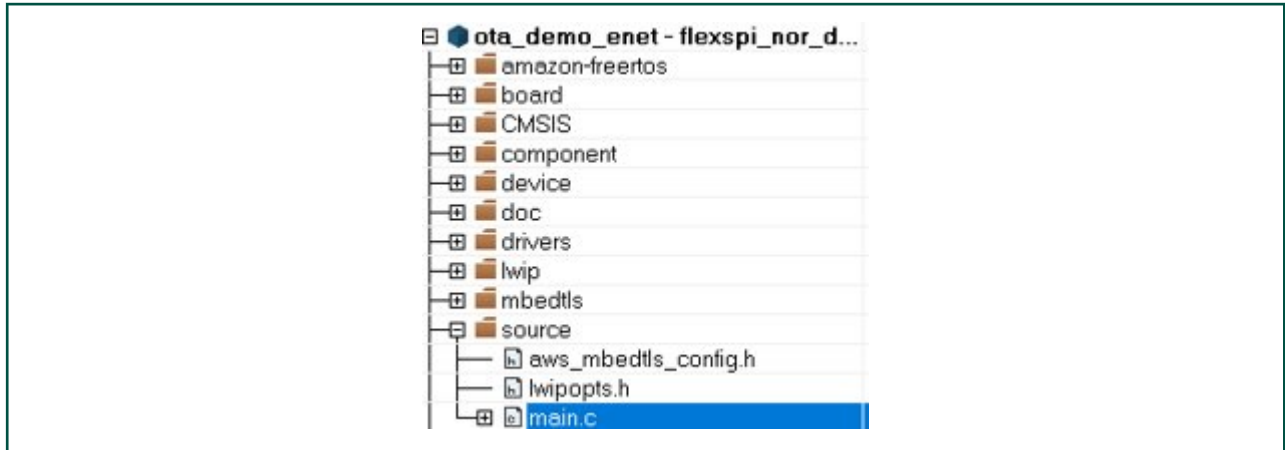
**Tags**  
Apply tags to your resources to help organize and identify them. A tag consists of a case-sensitive key-value pair.  

Tag name	Value	
<input type="text" value="Provide a tag name, e.g. Manufacturer"/>	<input type="text" value="Provide a tag value, e.g. Acme-Corporation"/>	<input type="button" value="Clear"/>
<input type="button" value="Add another"/>		

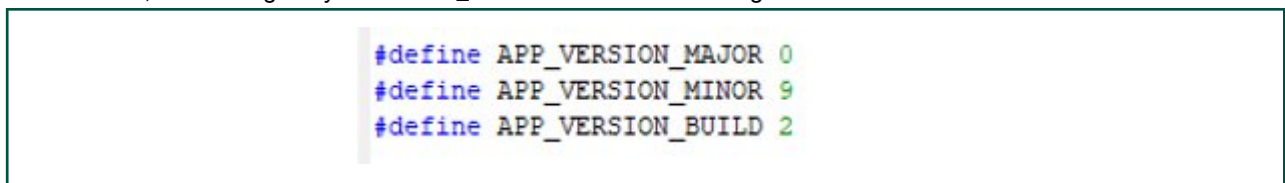
## 16. Before clicking Create the application needs to run

## 6.4 Running the application

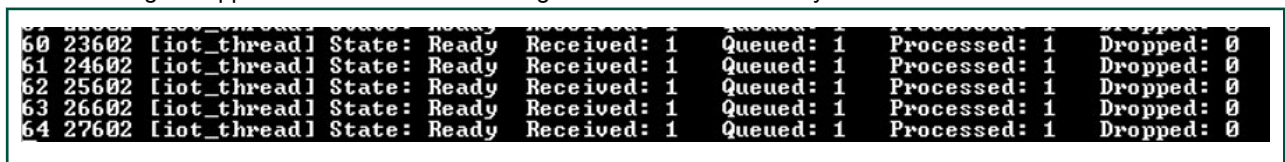
1. Open main.c.



- Go to line 86, then change any of the APP\_VERSION macro to the original value.



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



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

CREATE JOB

Create an Amazon FreeRTOS OTA update job

Job type

A job can run on the devices and/or groups selected, or remain open, and apply to devices later added to a group.

☒ Your job will complete after deploying to the selected devices/groups (snapshot)
 ☐ Your job will continue deploying to any devices added to the selected groups (continuous)

ID

OTAUpdateJob

Description (optional)

Give your job a helpful description

Tags

Apply tags to your resources to help organize and identify them. A tag consists of a case-sensitive key-value pair.

Tag name

Value

Clear

Add another

Cancel

Back

Create

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

```

55 18633 [iot_thread] State: Ready Received: 1 Queued: 1 Processed: 1 Dropped: 0
56 19633 [iot_thread] State: Ready Received: 1 Queued: 1 Processed: 1 Dropped: 0
57 20633 [iot_thread] State: Ready Received: 1 Queued: 1 Processed: 1 Dropped: 0
59 21291 [OTA Task] [prvParseJSONbyModel] Extracted parameter [ streamname: AFR_OTA-906e2011-a543-460 21300 [OTA Task]
[prvParseJSONbyModel] Extracted parameter [ filepath: /device/updates ]
61 21308 [OTA Task] [prvParseJSONbyModel] Extracted parameter [ filesize: 260608 ]
62 21315 [OTA Task] [prvParseJSONbyModel] Extracted parameter [ fileid: 0 ]
63 21322 [OTA Task] [prvParseJSONbyModel] Extracted parameter [ certfile: /certificates/authcert.pem 21321 [OTA Task]
[prvParseJSONbyModel] Extracted parameter [ sig-sha256-ecdsa: MEUCIQC2HsafgBckf65 21340 [OTA Task] [prvParseJobDoc] J
Job was accepted. Attempting to start transfer.
66 21342 [OTA Task] [INFO [MQTT]] [MQTT connection 2020b468, SUBSCRIBE operation 2020b7b8] Wait 21445 [OTA Task]
67 21358 [OTA Task] [INFO [MQTT]] [MQTT connection 2020b468, SUBSCRIBE operation 2020b7b8] Wait 21454 [OTA Task] [prvSubscribeToDataS
[INFO [MQTT]] [MQTT connection 2020b468, SUBSCRIBE operation 2020b7b8] Wait 21454 [OTA Task] [prvSubscribeToDataS

```

7. Start file transfer.

```

77 24265 [OTA Task] [OTA-NXP] WriteBlock 0 : 400
78 24269 [OTA Task] [prvIngestDataBlock] Remaining: 254
79 24308 [OTA Task] [prvIngestDataBlock] Received file block 1, size 1024

```

```

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/APR_OTA-900
1 [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=0x00003fe0, image[1].size=0x00003fe0

```

## 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...
10 4956 [iot_thread] ...

```

## 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](http://nxp.com)

### **Web Support:**

[nxp.com/support](http://nxp.com/support)

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: [nxp.com/SalesTermsandConditions](http://nxp.com/SalesTermsandConditions).

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX, EMBRACE, GREENCHIP, HITAG, ICODE, JCOP, LIFE VIBES, MIFARE, MIFARE CLASSIC, MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLX, SMARTMX, STARPLUG, TOPFET, TRENCHMOS, UCODE, Freescale, the Freescale logo, AltiVec, CodeWarrior, ColdFire, ColdFire+, the Energy Efficient Solutions logo, Kinetis, Layerscape, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QorIQ, QorIQ Qonverge, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, Tower, TurboLink, EdgeScale, EdgeLock, eIQ, and Immersive3D are trademarks of NXP B.V. All other product or service names are the property of their respective owners. AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro,  $\mu$ Vision, Versatile are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

© NXP B.V. 2019-2020.

All rights reserved.

For more information, please visit: <http://www.nxp.com>

For sales office addresses, please send an email to: [salesaddresses@nxp.com](mailto:salesaddresses@nxp.com)

Date of release: 15 June 2020  
Document identifier: AWSOTAUG

