

User Manual

DA16200 DA16600 Getting Started with AWS® IoT Core

Abstract

The DA16200/DA16600 is a highly integrated ultra-low power Wi-Fi system on chip (SoC) that allows users to develop a complete Wi-Fi solution on a single chip. This document is a DA16200/DA16600 getting started guide intended to help new or existing developers quickly get started using AWS® IoT Core.

Contents

Abstract	1
Contents	2
Figures.....	3
1 Terms and Definitions.....	6
2 References	6
3 AWS IoT	7
3.1 Configure AWS IoT	7
3.1.1 Sign Up for AWS Account.....	7
3.1.2 Connect Devices to AWS IoT	8
3.1.3 Configure Amazon Cognito	31
3.1.4 Set Up AWS IAM	39
3.1.5 Create Amazon S3 Bucket	41
4 Door Lock Reference Application	42
4.1 Reference Application in DA16200/DA16600	42
4.1.1 Open Door	42
4.1.2 Close Door.....	44
4.2 Reference Application in Host MCU	47
4.2.1 Download Package for Door Lock Reference Application in Host MCU	47
4.2.2 Hardware Connections between DA16200/DA16600 and Host MCU	48
4.2.3 Programming Firmware Images for DA16200/DA16600.....	52
4.2.4 Configure Components for Testing.....	54
4.2.5 Test without Host MCU.....	56
4.2.6 Test with Host MCU.....	57
4.3 Mobile App Demo.....	62
4.3.1 Open Door	63
4.3.2 Close Door.....	64
5 OTA Update.....	66
5.1 Create S3 Bucket	66
5.2 Upload Image File and JSON File	73
5.3 Create Job.....	74
5.4 Execute OTA Update	78
Appendix A Provisioning.....	81
A.1 Android Application	81
Appendix B AT Commands for AWS IoT	84
B.1 Operating Modes.....	84
B.1.1 Setting Mode.....	84
B.1.2 Provisioning Mode	85
B.1.3 Communication Mode.....	85
B.2 Configuring Topic to Publish, Subscribe and Shadow	86
B.2.1 Configure Topics.....	86
B.3 AT Command List	87
B.3.1 Basic Set.....	87

DA16200 DA16600 Getting Started with AWS® IoT Core

B.3.2	TLS Certificate	87
B.3.3	PIN MUX	88
B.3.4	Configure Data as Topics	89
B.3.5	Command – MCU to DA16200/DA16600	89
B.3.6	Command – DA16200/DA16600 to MCU	90
B.3.7	DA16200/DA16600 Status – DA16200/DA16600 to MCU	90
Appendix C Troubleshooting		91
C.1	Operational Issue	91
Revision History		92

Figures

Figure 1: Sign Up for AWS Account	7
Figure 2: Register Things	8
Figure 3: Create Single Thing	9
Figure 4: Thing Name	10
Figure 5: Thing without Certificate	11
Figure 6: Created Thing	11
Figure 7: Classic Shadow	12
Figure 8: Device Shadow Document	13
Figure 9: Create Certificates	14
Figure 10: Create Certificates (Continued)	14
Figure 11: Download Certificates and Keys	15
Figure 12: Activate Certificate	16
Figure 13: Create Policy	16
Figure 14: Enter JSON Policy Statement	18
Figure 15: Created Policy	19
Figure 16: Check Created Policy	19
Figure 17: Attach Policy to Certificate	20
Figure 18: Attach Policy	21
Figure 19: Attach Things to Certificate	22
Figure 20: Attach to Thing	23
Figure 21: Create Rule	23
Figure 22: Specify Rule Properties	24
Figure 23: Configure SQL Statement	25
Figure 24: Attach Rule Actions	26
Figure 25: Attach Rule Actions (Continued)	27
Figure 26: Create IAM Role to Save Log Files	27
Figure 27: Review Rules	28
Figure 28: Created Act Rule	29
Figure 29: Created Act Role	29
Figure 30: Attach Policy to Role	29
Figure 31: AWSIoTFullAccess Policy	30
Figure 32: Attached Policies	30
Figure 33: Create User Pool	31
Figure 34: Configure Sign-in Options	32
Figure 35: Configure Security Requirements	33
Figure 36: Configure Security Requirements (Continued)	34
Figure 37: Configure Message Delivery	35
Figure 38: Integrate App Client	36
Figure 39: Created User Pool	37
Figure 40: Create Identity Pool	37
Figure 41: Create Identity Pool Trust	37
Figure 42: Configure Permissions	38

DA16200 DA16600 Getting Started with AWS® IoT Core

Figure 43: Configure Properties	38
Figure 44: Created Identity Pools	39
Figure 45: IAM Role	39
Figure 46: Attach Policies	40
Figure 47: AWSIoTFullAccess Policy	40
Figure 48: AmazonS3FullAccess Policy	40
Figure 49: Attached Policies	41
Figure 50: Architecture of AWS IoT	42
Figure 51: Message Flows of Opening Door	43
Figure 52: Open Dooring on Mobile App	43
Figure 53: Shadow State When Door is Open	44
Figure 54: Message Flows of Closing Door	45
Figure 55: Closing Door on Mobile App	45
Figure 56: Shadow State when Door is Closed	46
Figure 57: AWS IoT Using Firmware Images for AT Commands and Host MCU	47
Figure 58: Hardware Configuration	48
Figure 59: Default UART HW Connection	49
Figure 60: Sample Example of UART1 Connection	50
Figure 61: HW Connection for Waking Up DA16200/DA16600	50
Figure 62: Default Pin Configuration for Waking Up Host MCU	51
Figure 63: Another Pin Configuration for Waking up Host MCU	51
Figure 64: Factory Reset Button on DA16200 EVB	52
Figure 65: Factory Reset Button on DA16600 EVB	52
Figure 66: e²studio Project File	57
Figure 67: FSP Configuration	58
Figure 68: Thing Name in MCU Source Code	58
Figure 69: Build Project	59
Figure 70: Debug Configurations	59
Figure 71: Set Debug Configurations	60
Figure 72: Opened Status on Application and AWS IoT Console	63
Figure 73: Closed Status on Application and AWS IoT Console	64
Figure 74: OTA Update	66
Figure 75: Create Bucket for OTA Update	66
Figure 76: Bucket Name for OTA Update	68
Figure 77: Created Buckets for OTA	68
Figure 78: Edit Bucket for Public Access	69
Figure 79: Public Access Settings for Bucket	69
Figure 80: Confirm Settings	70
Figure 81: Settings Updated	70
Figure 82: Public Access for Everyone	71
Figure 83: Bucket Policy Editor	72
Figure 84: Bucket Policy JSON	72
Figure 85: Upload Files	73
Figure 86: Ready to Upload	73
Figure 87: URL of Source	74
Figure 88: Uploaded Files	74
Figure 89: Completed Setup for OTA Update	74
Figure 90: Create Job	75
Figure 91: Create Custom Job	75
Figure 92: Make Job Name	76
Figure 93: Select Thing for OTA Update	76
Figure 94: Select JSON for OTA Update	77
Figure 95: Job Run Type	77
Figure 96: Job Being Created	78
Figure 97: Successful Created Job	78
Figure 98: Successful Job for OTA Update in Mobile App	79
Figure 99: Execute OTA Update in Android App	80
Figure 100: Provisioning Flow	81
Figure 101: Provisioning from Mobile App	83

DA16200 DA16600 Getting Started with AWS® IoT Core

Figure 102: Running AWS IoT Application from Mobile App 84

Figure 103: Setting Mode 85

Figure 104: Provisioning Mode..... 85

Figure 105: Communication Mode 86

Figure 106: Communication between MCU and Phone 86

1 Terms and Definitions

AP	Access Point
API	Application Programming Interface
AWS	Amazon Web Services
DPM	Dynamic Power Management
DTIM	Delivery Traffic Indication Map
IoT	Internet of Things
MCU	Micro-Controller Unit
OTA	Over The Air
SDK	Software Development Kit
TIM	Traffic Indication Map

2 References

- [1] DA16200MOD, Datasheet, Renesas Electronics
- [2] DA16600MOD, Datasheet, Renesas Electronics
- [3] UM-WI-056, DA16200 DA16600, FreeRTOS Getting Started Guide, User Manual, Renesas Electronics
- [4] UM-WI-042, DA16200 DA16600, Provisioning Mobile App, User Manual, Renesas Electronics

DA16200 DA16600 Getting Started with AWS® IoT Core

3 AWS IoT

The DA16200MOD/DA16600MOD is a full offload SoC for IoT applications such as security system, door lock, and smart applications. This section provides how to configure AWS IoT for communicating with the DA16200/DA16600 IoT device.

3.1 Configure AWS IoT

To connect a device to the AWS IoT server, the following components are required. This section describes how to set up requirements before using AWS IoT.

To configure AWS IoT server, follow the steps below:

- Set up AWS account and permissions
- Connect devices to AWS IoT
- Configure Amazon Cognito user pools and identity pools
- Set up Amazon IAM
- Create S3 bucket

3.1.1 Sign Up for AWS Account

To create an AWS account and grant permissions, follow the steps below:

1. Go to AWS website and create a free account (<https://portal.aws.amazon.com/billing/signup#/start/email>).
2. Create an administrative user for performing daily administrative tasks.
3. Open the AWS IoT console to get started with AWS IoT.

The screenshot shows the AWS 'Sign in' page. Under the 'Sign in' heading, there are two options: 'Root user' (unselected) and 'IAM user' (selected). The 'IAM user' option is highlighted with a blue border. Below these options, there is a text input field for 'Account ID (12 digits) or account alias' containing the text 'renesas_test_user_01'. A blue 'Next' button is positioned below the input field. At the bottom of the page, there is a link for 'New to AWS?' and a button labeled 'Create a new AWS account'.

Figure 1: Sign Up for AWS Account

NOTE

If a user does not have an AWS account, the user can test it with Renesas Electronics' test account. Contact Renesas Electronics for test account and password to sign in.

DA16200 DA16600 Getting Started with AWS® IoT Core

3.1.2 Connect Devices to AWS IoT

Users can configure and manage the thing objects, certificates, rules, jobs, policies, and other elements of IoT solutions through AWS IoT console. Prior to send data to and receive data from AWS IoT server, users should register a device first.

3.1.2.1 Register a Device in Thing Registry

In the Thing Registry, the devices connected to the AWS IoT server are represented by Things. The Thing Registry allows keeping records of all devices that are connected to an AWS IoT account. To register a device in the Thing Registry, complete the following steps.

1. On the AWS IoT console, expand **Registry** on the navigation pane.
2. Select **Manage > Things > Create things**. See [Figure 2](#).

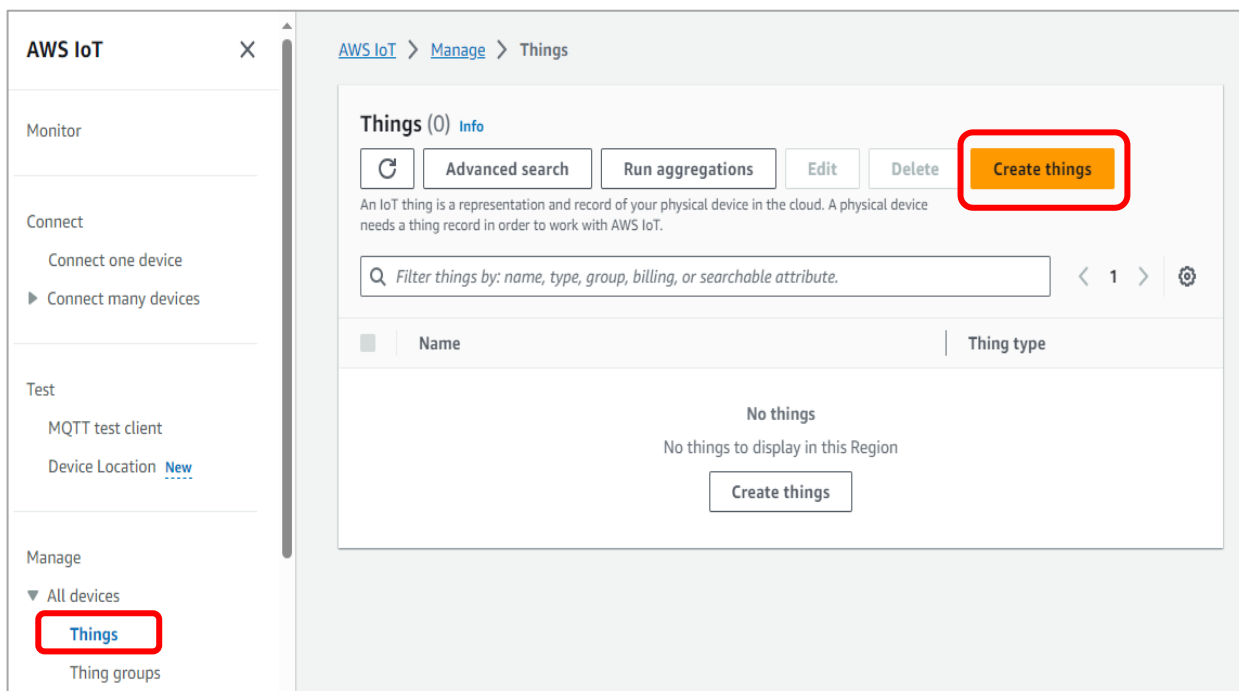


Figure 2: Register Things

3. Click **Create single thing**. See [Figure 3](#).

DA16200 DA16600 Getting Started with AWS® IoT Core

AWS IoT > Manage > Things > Create things

Create things [Info](#)

A thing resource is a digital representation of a physical device or logical entity in AWS IoT. Your device or entity needs a thing resource in the registry to use AWS IoT features such as Device Shadows, events, jobs, and device management features.

Number of things to create

☒ **Create single thing**
Create a thing resource to register a device. Provision the certificate and policy necessary to allow the device to connect to AWS IoT.

☐ **Create many things**
Create a task that creates multiple thing resources to register devices and provision the resources those devices require to connect to AWS IoT.

Cancel **Next**

Figure 3: Create Single Thing

4. In the **Thing name** field, enter a device name, such as “MyTestDoorLock”, and click **Unnamed shadow (classic)** and **Next** to add the device to the Thing Registry. See [Figure 4](#).

DA16200 DA16600 Getting Started with AWS® IoT Core

AWS IoT > Manage > Things > Create things > Create single thing

Step 1
Specify thing properties

Step 2 - optional
Configure device certificate

Step 3 - optional
Attach policies to certificate

Specify thing properties [Info](#)

A thing resource is a digital representation of a physical device or logical entity in AWS IoT. Your device or entity needs a thing resource in the registry to use AWS IoT features such as Device Shadows, events, jobs, and device management features.

Thing properties [Info](#)

Thing name

MyTestDoorLock

Enter a unique name containing only: letters, numbers, hyphens, colons, or underscores. A thing name can't contain any spaces.

Additional configurations

You can use these configurations to add detail that can help you to organize, manage, and search your things.

- ▶ Thing type - optional
- ▶ Searchable thing attributes - optional
- ▶ Thing groups - optional
- ▶ Billing group - optional
- ▶ Packages and versions - optional

Device Shadow [Info](#)

Device Shadows allow connected devices to sync states with AWS. You can also get, update, or delete the state information of this thing's shadow using either HTTPs or MQTT topics.

☐ No shadow

☐ Named shadow
Create multiple shadows with different names to manage access to properties, and logically group your devices properties.

☒ **Unnamed shadow (classic)**
A thing can have only one unnamed shadow.

▶ Edit shadow statement - optional

Cancel **Next**

Figure 4: Thing Name

5. Click **Skip a creating a certificate at this time**. See [Figure 5](#).

DA16200 DA16600 Getting Started with AWS® IoT Core

AWS IoT > Manage > Things > Create things > Create single thing

Step 1
[Specify thing properties](#)

Step 2 - optional
Configure device certificate

Configure device certificate - optional

A device requires a certificate to connect to AWS IoT. You can choose how to register a certificate for your device now, or you can create and register a certificate for your device later. Your device won't be able to connect to AWS IoT until it has an active certificate with an appropriate policy.

Device certificate

☐ Auto-generate a new certificate (recommended)
Generate a certificate, public key, and private key using AWS IoT's certificate authority.

☐ Use my certificate
Use a certificate signed by your own certificate authority.

☐ Upload CSR
Register your CA and use your own certificates on one or many devices.

☒ Skip creating a certificate at this time
You can create a certificate for this thing and attach a policy to the certificate at a later time.

Cancel Previous **Create thing**

Figure 5: Thing without Certificate

The Thing created to perform the test is named as **MyTestDoorLock**.

- Click the created Thing. See [Figure 6](#).

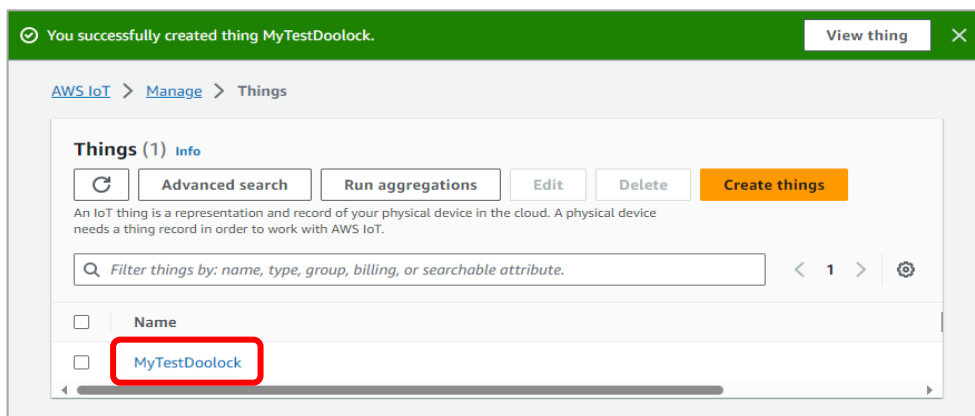


Figure 6: Created Thing

- For the shadow function of the thing, select the **Things** and **Device Shadows** and then click **Classic shadow**. See [Figure 7](#) and [Figure 8](#).

8.

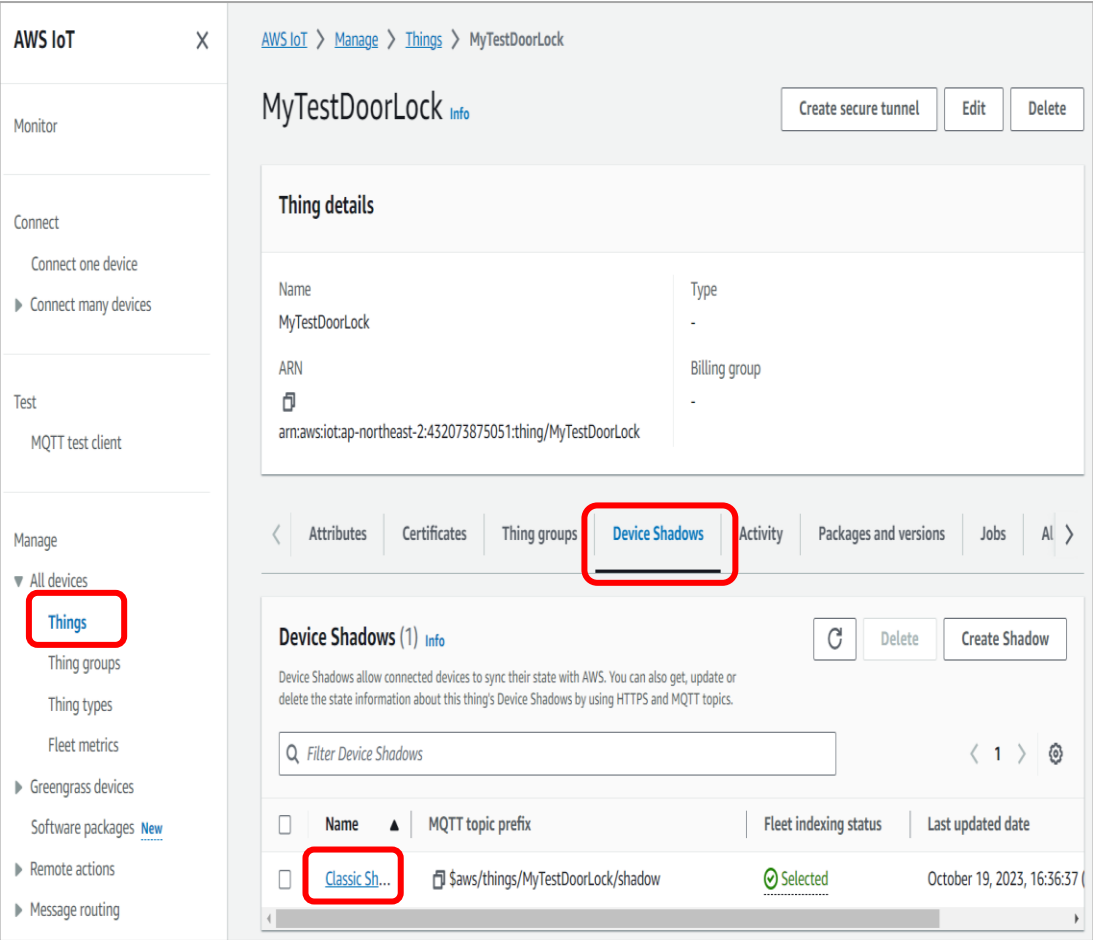
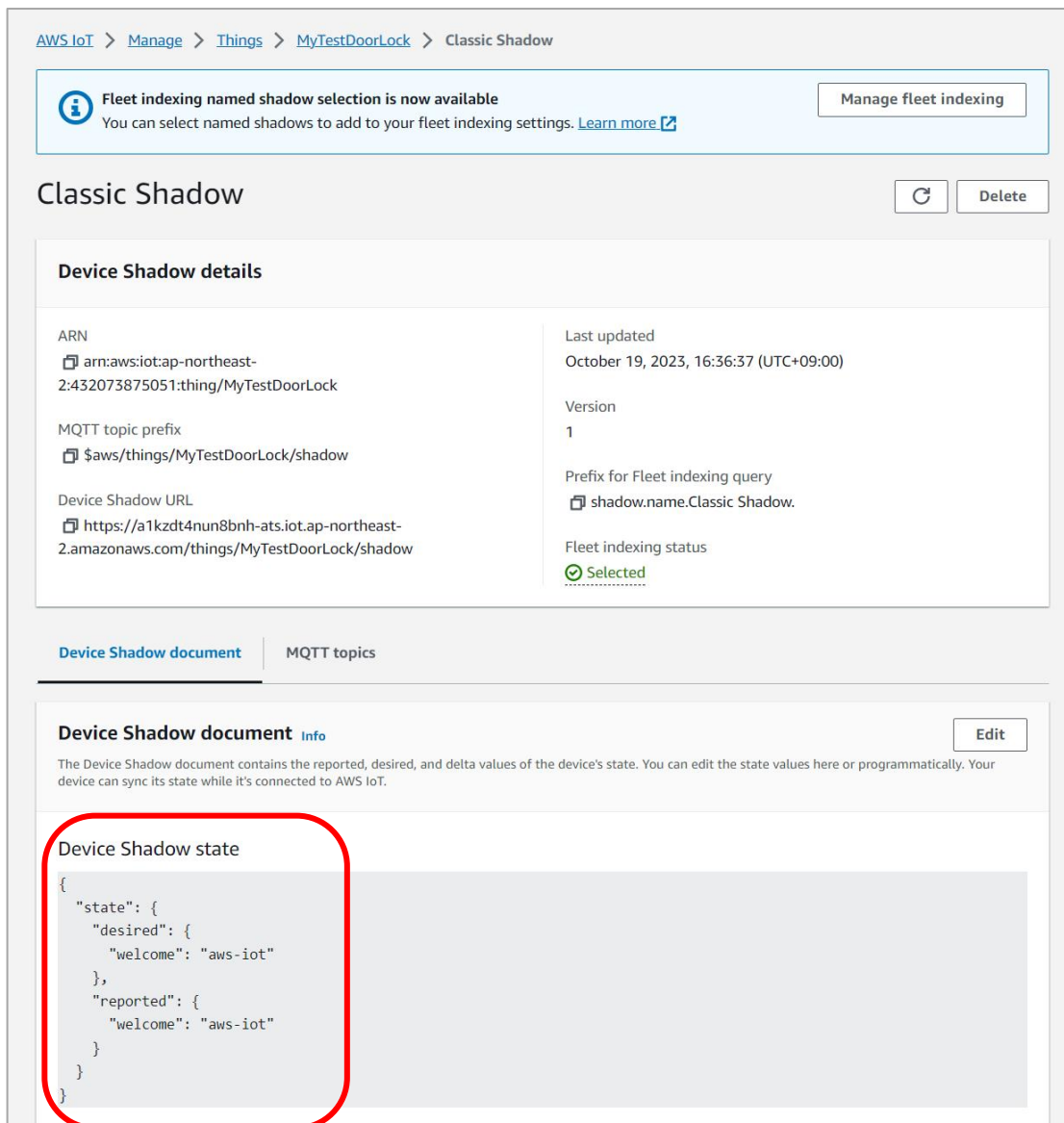


Figure 7: Classic Shadow

DA16200 DA16600 Getting Started with AWS® IoT Core



[AWS IoT](#) > [Manage](#) > [Things](#) > [MyTestDoorLock](#) > Classic Shadow

Fleet indexing named shadow selection is now available
 You can select named shadows to add to your fleet indexing settings. [Learn more](#)

[Manage fleet indexing](#)

Classic Shadow

[Refresh](#) [Delete](#)

Device Shadow details

ARN arn:aws:iot:ap-northeast-2:432073875051:thing/MyTestDoorLock	Last updated October 19, 2023, 16:36:37 (UTC+09:00)
MQTT topic prefix \$aws/things/MyTestDoorLock/shadow	Version 1
Device Shadow URL https://a1kzdt4nun8bnh-ats.iot.ap-northeast-2.amazonaws.com/things/MyTestDoorLock/shadow	Prefix for Fleet indexing query shadow.name.Classic Shadow.
	Fleet indexing status Selected

[Device Shadow document](#) [MQTT topics](#)

Device Shadow document [Info](#)

[Edit](#)

The Device Shadow document contains the reported, desired, and delta values of the device's state. You can edit the state values here or programmatically. Your device can sync its state while it's connected to AWS IoT.

Device Shadow state

```

{
  "state": {
    "desired": {
      "welcome": "aws-iot"
    },
    "reported": {
      "welcome": "aws-iot"
    }
  }
}
  
```

Figure 8: Device Shadow Document

For more information on device shadows for AWS IoT, visit AWS IoT Device Shadow service (<https://docs.aws.amazon.com/iot/latest/developerguide/iot-device-shadows.html>).

3.1.2.2 Create and Activate Device Certificate

The communication between the device and the AWS IoT web service is protected by X.509 certificates. The user can let the AWS IoT generate a certificate or the user can use their own X.509 certificate. This section shows that AWS IoT generates the X.509 certificate.

The certificates should be activated before use. Complete the following steps to create and activate a device certificate.

DA16200 DA16600 Getting Started with AWS® IoT Core

1. On the navigation pane, select **Secure**, **Certificates** (as necessary) and then click **Add certificate** and **Create certificate**. See Figure 9.

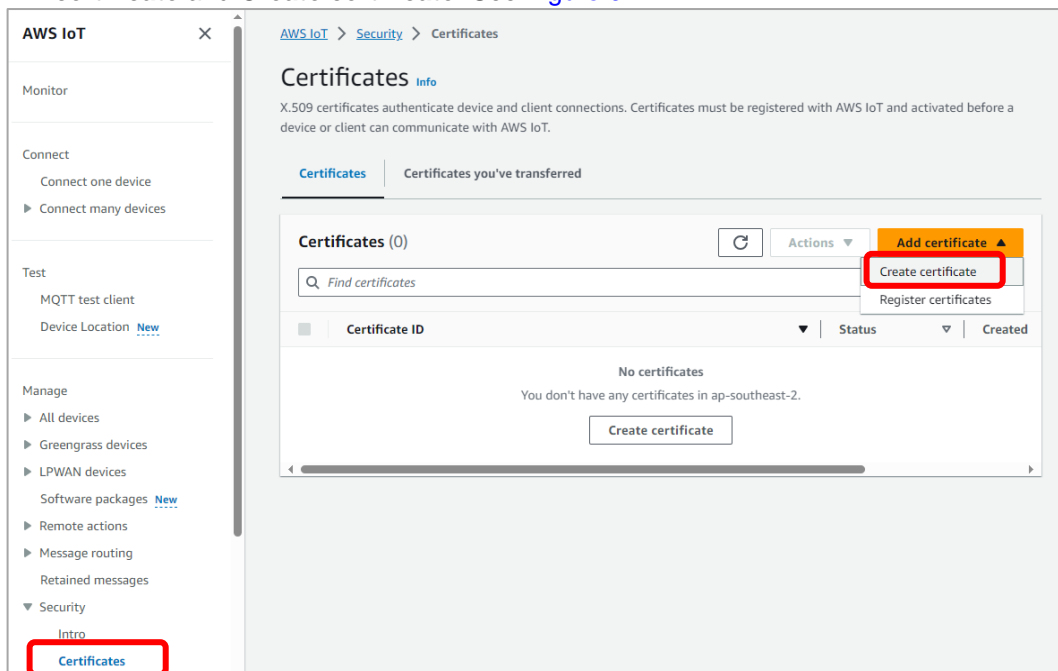


Figure 9: Create Certificates

2. Select **Auto-generate new certificate (recommended)** and **Activate**. See Figure 10.

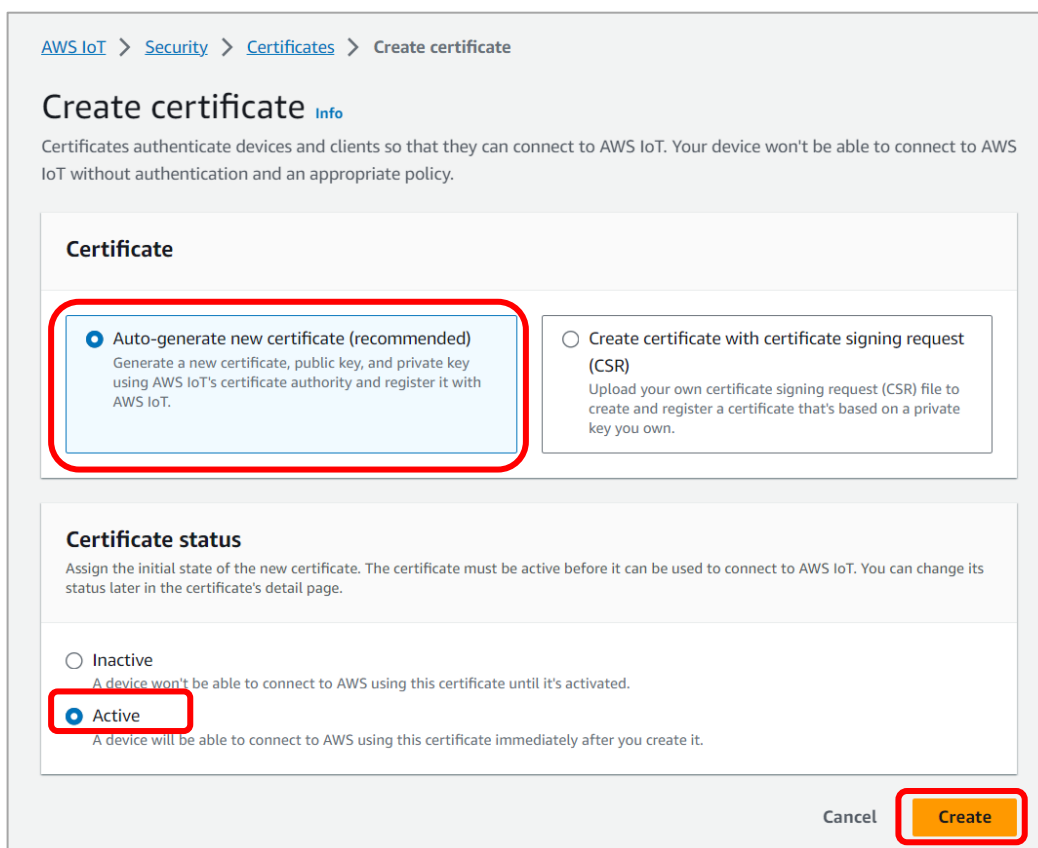


Figure 10: Create Certificates (Continued)

DA16200 DA16600 Getting Started with AWS® IoT Core

- There are three required certificates to download. On the **Certificate Created** page, click **Download** to download the device certificate, private key, and root CA certificates for AWS IoT, and then save the downloads to your computer. The certificate files should be saved before leaving this page. After leaving this page in the console, users no longer have access to the certificate files. Renesas recommends that Device certificate, Private key file, and Root CA should be downloaded in sequential order. See [Figure 11](#).

Download certificates and keys

Download certificates and keys

Download and install the certificate and key files to your device so that it can connect securely to AWS IoT. You can download the certificate now, or later, but the key files can only be downloaded now.

Device certificate

9e43e7e6594...te.pem.crt

Download

Key files

The key files are unique to this certificate and can't be downloaded after you leave this page. Download them now and save them in a secure place.

This is the only time you can download the key files for this certificate.

Public key file

9e43e7e659461f0b979c217...9c9432a-public.pem.key

Download

Private key file

9e43e7e659461f0b979c217...c9432a-private.pem.key

Download

Root CA certificates

Download the root CA certificate file that corresponds to the type of data endpoint and cipher suite you're using. You can also download the root CA certificates later.

Amazon trust services endpoint

RSA 2048 bit key: Amazon Root CA 1

Download

Amazon trust services endpoint

ECC 256 bit key: Amazon Root CA 3

Download

If you don't see the root CA certificate that you need here, AWS IoT supports additional root CA certificates. These root CA certificates and others are available from our developer guides.

Continue

Figure 11: Download Certificates and Keys

For Root CA, visit the AWS Docs site (<https://docs.aws.amazon.com/iot/latest/developerguide/server-authentication.html#server-authentication-certs>). Root CA certificates are subjected to expiration and/or revocation.

User Manual

Revision 1.5

Jan. 26, 2024

CFR0012

15 of 94

© 2024 Renesas Electronics

DA16200 DA16600 Getting Started with AWS® IoT Core

4. The certificate should show as **Active** in the list of certificates. See Figure 12.

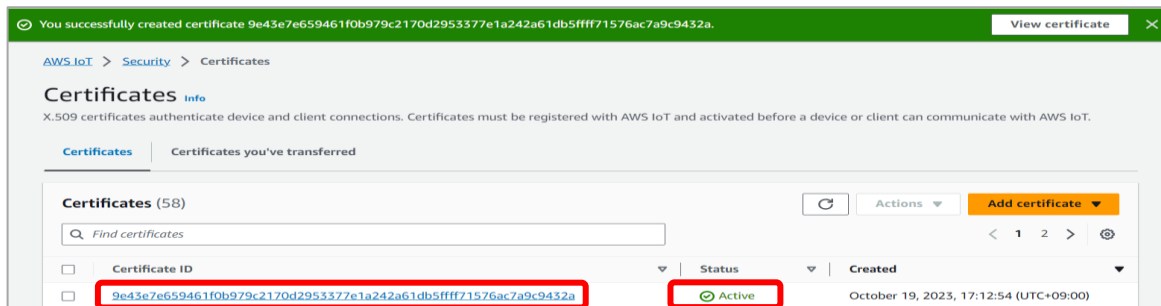


Figure 12: Activate Certificate

3.1.2.3 Create Policy

The X.509 certificates are used to authenticate the device with the AWS IoT. The AWS IoT policies are used to authorize the device for AWS IoT operations, such as subscribing or publishing to MQTT topics. The device displays its certificate only while connecting to the AWS IoT. To allow the device for AWS IoT operations, the user should create an AWS IoT policy and attach that policy to the device certificate.

To create an AWS IoT policy, complete the following steps:

1. On the navigation pane, expand **Secure** and click **Policies**. See Figure 13.
2. Click **Create policy**.

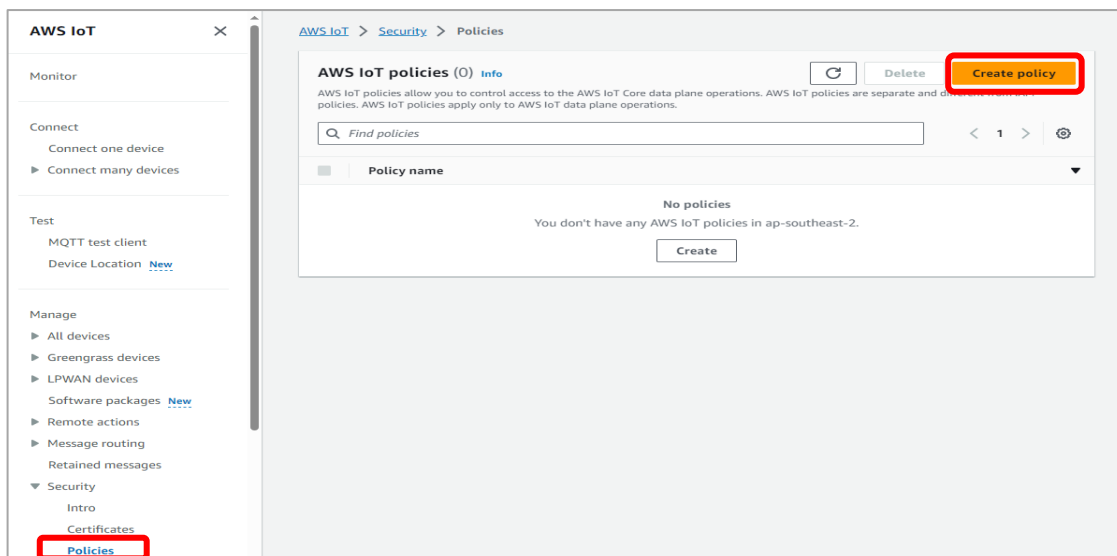


Figure 13: Create Policy

3. On the Create policy page:
 - a. In the **Policy name** field under Policy properties section, enter a name for the policy (for example, MyTestPolicy). Renesas strongly recommends not using personally identifiable information in policy names.
 - b. In the **Policy document** section, select JSON, and then copy and paste the following json statement.

DA16200 DA16600 Getting Started with AWS® IoT Core

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

- c. After entering the required information, choose **Create**. See [Figure 14](#).

NOTE

The examples in this document are intended only for development environments. All devices in your production fleet must have credentials with privileges that authorize only intended actions on specific resources. The specific permission policies may vary depending on use cases. Identify the permission policies that best meet the business and security requirements. For more information, see Example policies and Security Best practices in AWS IoT.

DA16200 DA16600 Getting Started with AWS® IoT Core

AWS IoT > Security > Policies > Create policy

Create policy Info

AWS IoT Core policies allow you to manage access to the AWS IoT Core data plane operations.

Policy properties

AWS IoT Core supports named policies so that many identities can reference the same policy document.

Policy name

MyTestPolicy

A policy name is an alphanumeric string that can also contain period (.), comma (,), hyphen(-), underscore (_), plus sign (+), equal sign (=), and at sign (@) characters, but no spaces.

► Tags - optional

Policy statements | Policy examples

Policy document Info

An AWS IoT policy contains one or more policy statements. Each policy statement contains actions, resources, and an effect that grants or denies the actions by the resources.

Builder JSON

Policy document

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Action": "iot:*",
7       "Resource": "*"
8     },
9     {
10      "Effect": "Allow",
11      "Action": "s3:*",
12      "Resource": "*"
13    }
14  ]
15 }
```

JSON Line 15, Column 2 Errors: 0 Warnings: 0

Cancel Create

Figure 14: Enter JSON Policy Statement

4. Select **Secure** > **Policies** to view the created policies. See [Figure 15](#).
5. Click the policy to view the details. An example of the selected policy content is shown in [Figure 16](#).

DA16200 DA16600 Getting Started with AWS® IoT Core

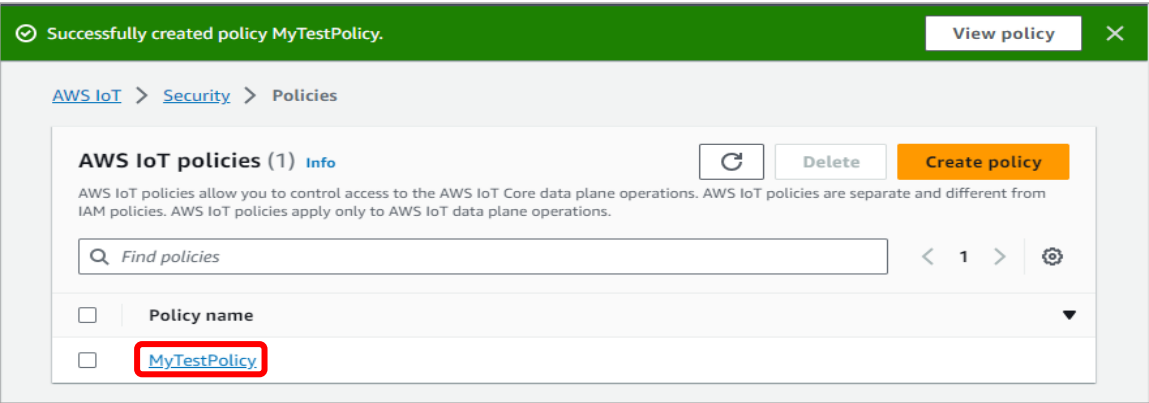


Figure 15: Created Policy

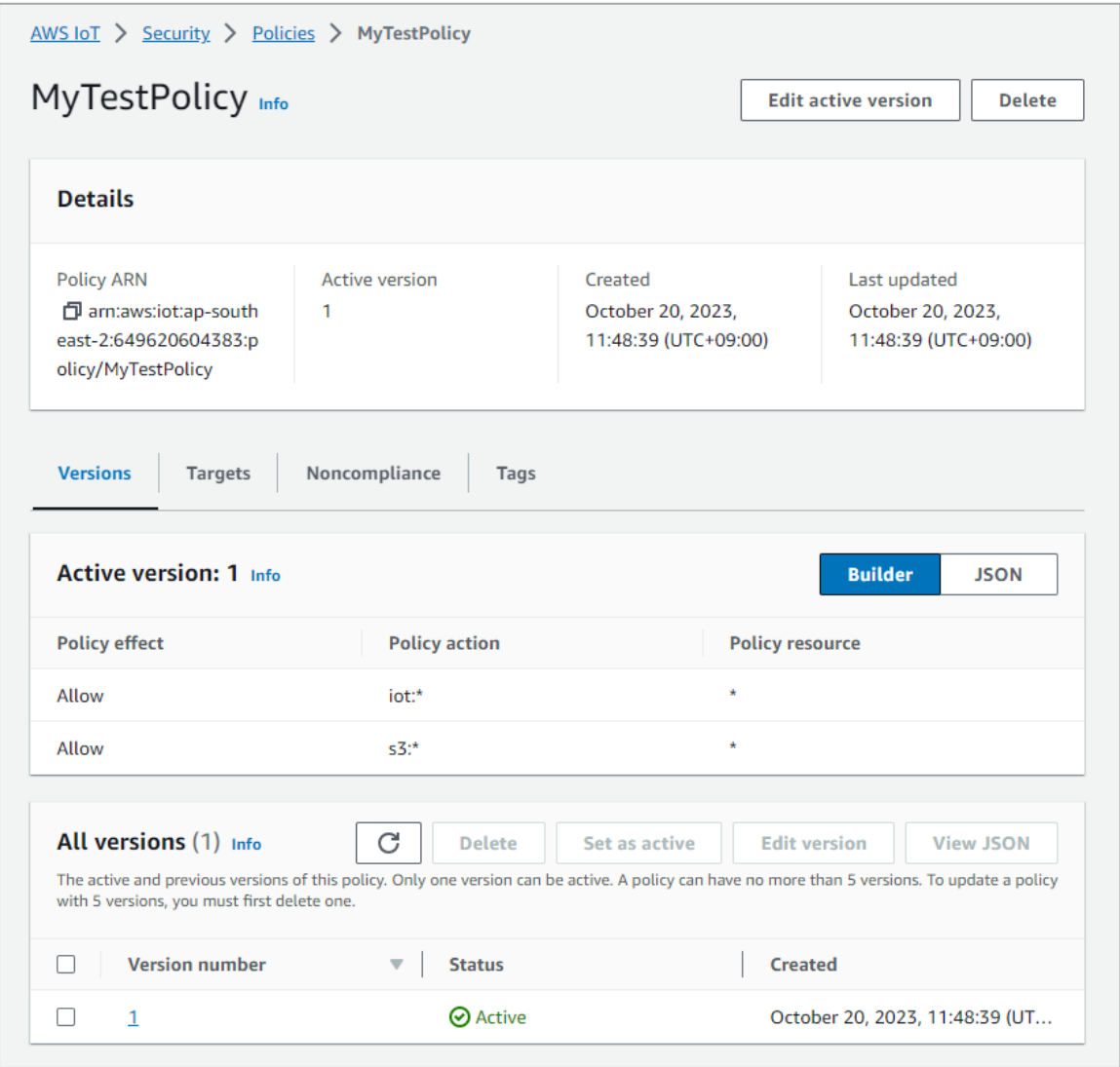


Figure 16: Check Created Policy

DA16200 DA16600 Getting Started with AWS® IoT Core

3.1.2.4 Attach Certificate to Thing and Policy

After an AWS IoT policy is created, the user must attach that policy to the device certificate. The attachment of an AWS IoT policy to a certificate gives the device the permissions that are specified in the policy. To attach the AWS IoT Policy to a device certificate, follow the steps below:

- 1. Go to the certificate created by the user, select **Policies**, and click **Attach Policy**. See Figure 17.

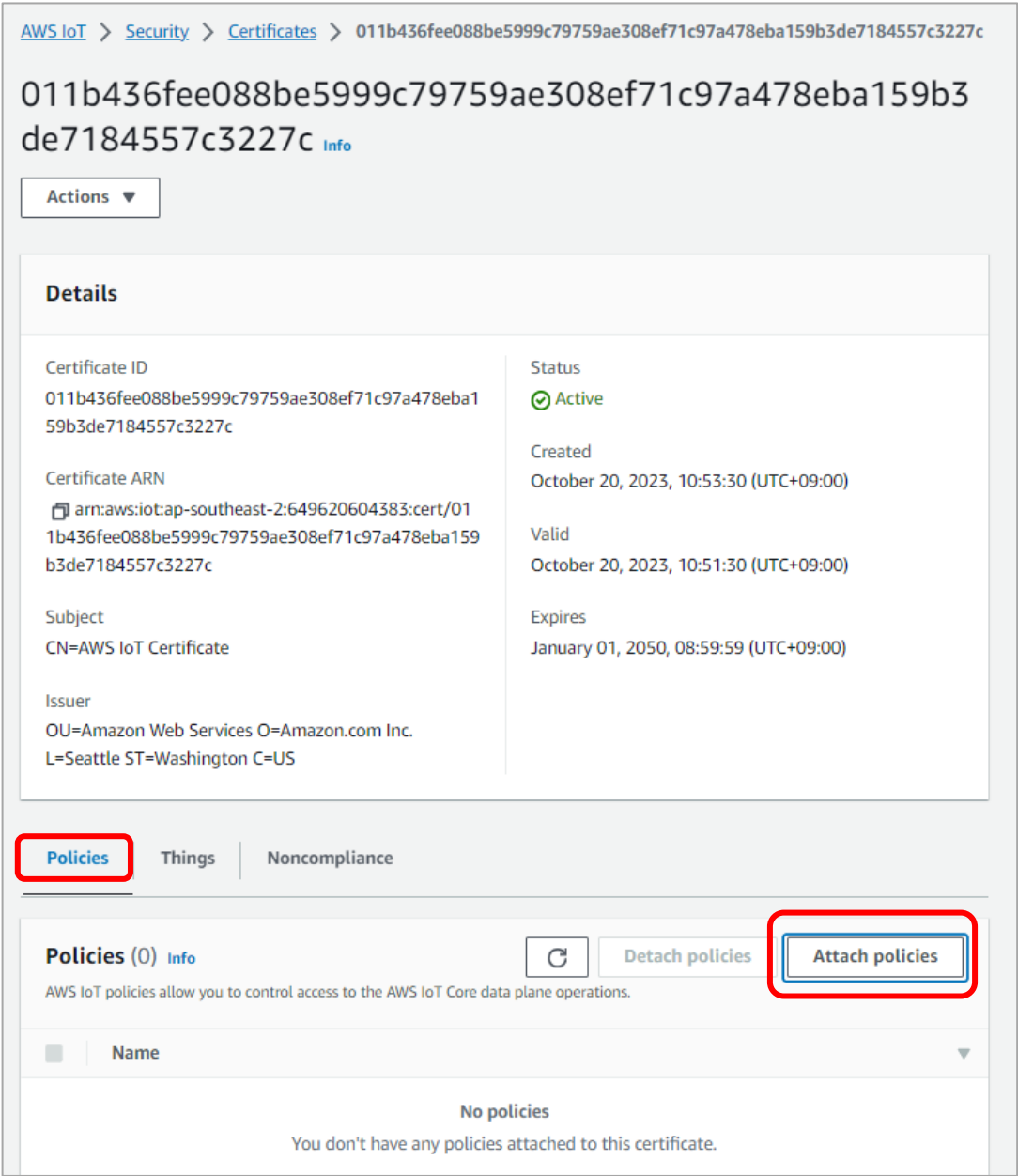


Figure 17: Attach Policy to Certificate

- 2. Select the checkbox of the created policy and click **Attach policies**. See Figure 18.

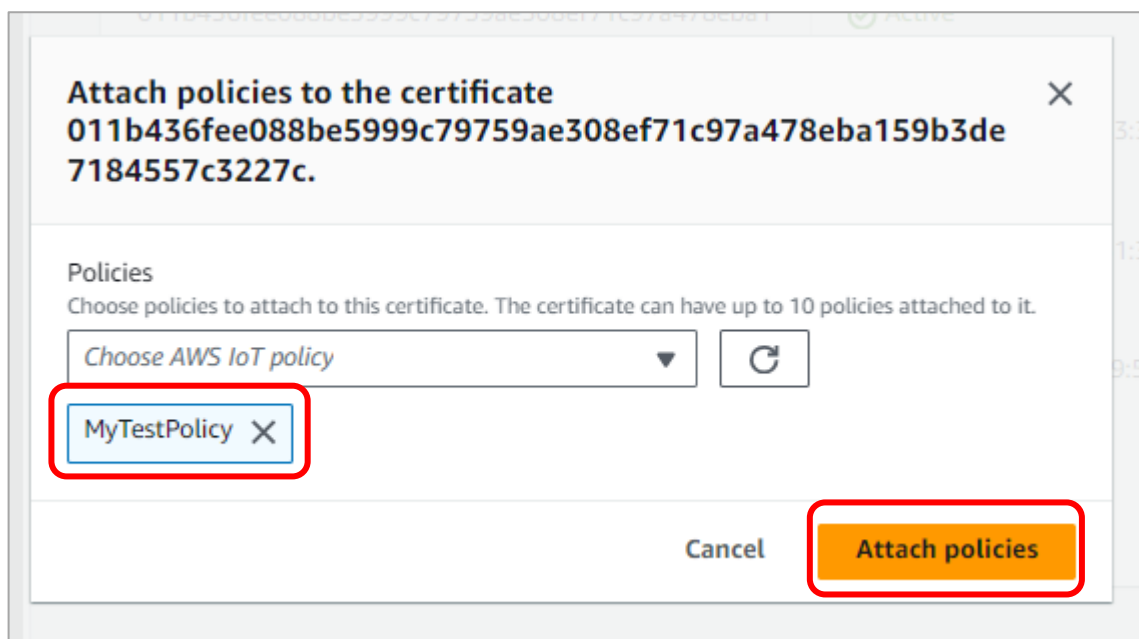


Figure 18: Attach Policy

A device should have a certificate, private key, and root CA certificate to authenticate with the AWS IoT. Renesas recommends that the user attaches the device certificate to the Thing that represents the device in AWS IoT. This allows the user to create AWS IoT policies that grant permissions based on certificates attached to Things.

3. Go to the certificate created by the user, select **Things** and click **Attach to things**. See [Figure 19](#).

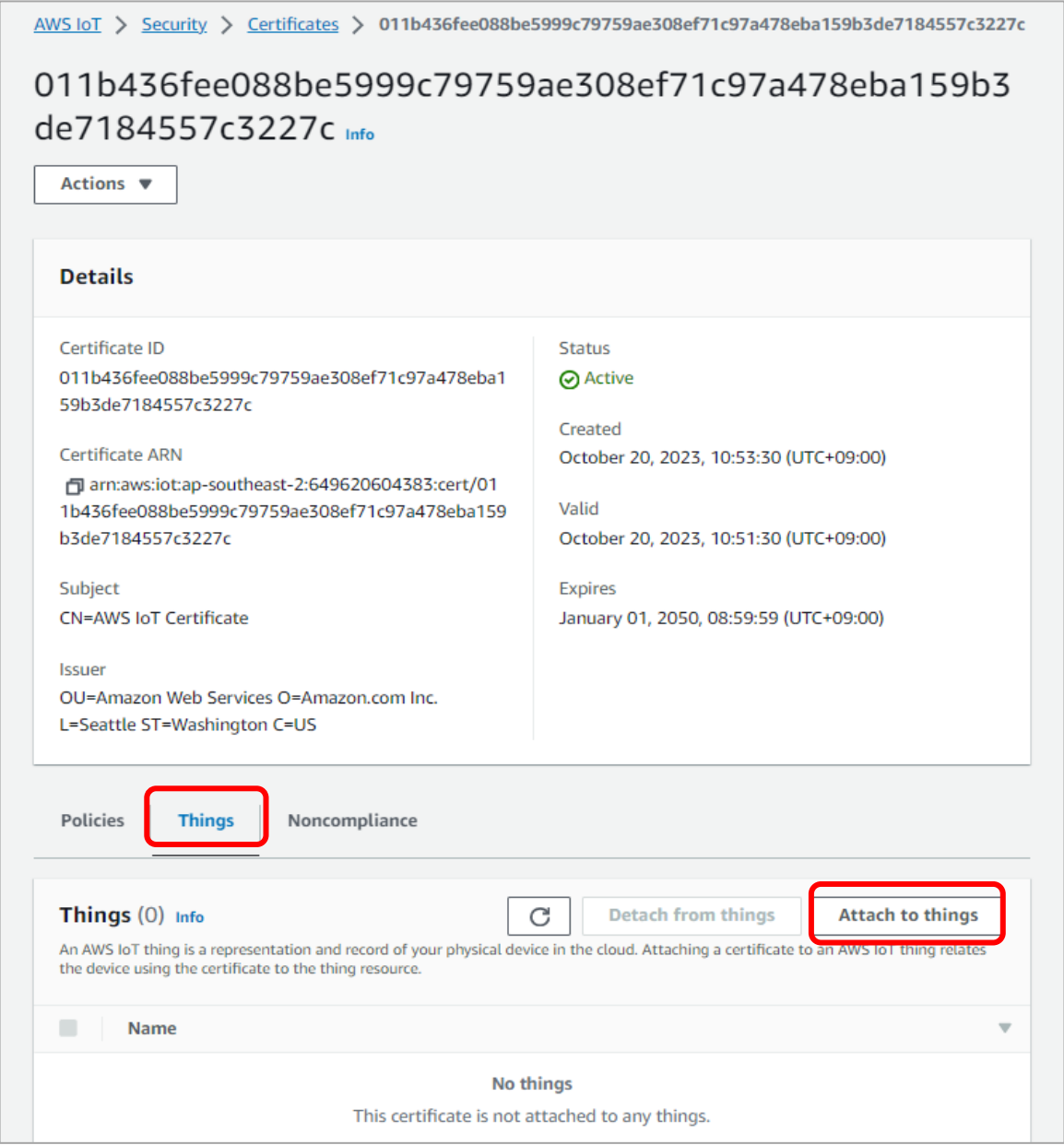


Figure 19: Attach Things to Certificate

4. Select the checkbox of the Thing that was created and click **Attach to thing**. See Figure 20.

DA16200 DA16600 Getting Started with AWS® IoT Core

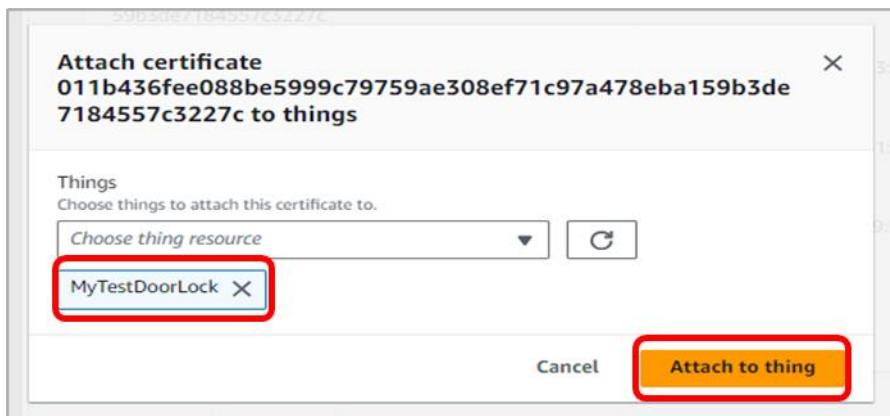


Figure 20: Attach to Thing

3.1.2.5 Store Events in S3 Bucket

Complete the following steps to store log files for the Door lock, and see 3.1.5 on how to create Amazon S3 bucket.

1. Select **AWS console** > **AWS IoT Core** > **Message Routing** > **Rules** > **Create rule**.

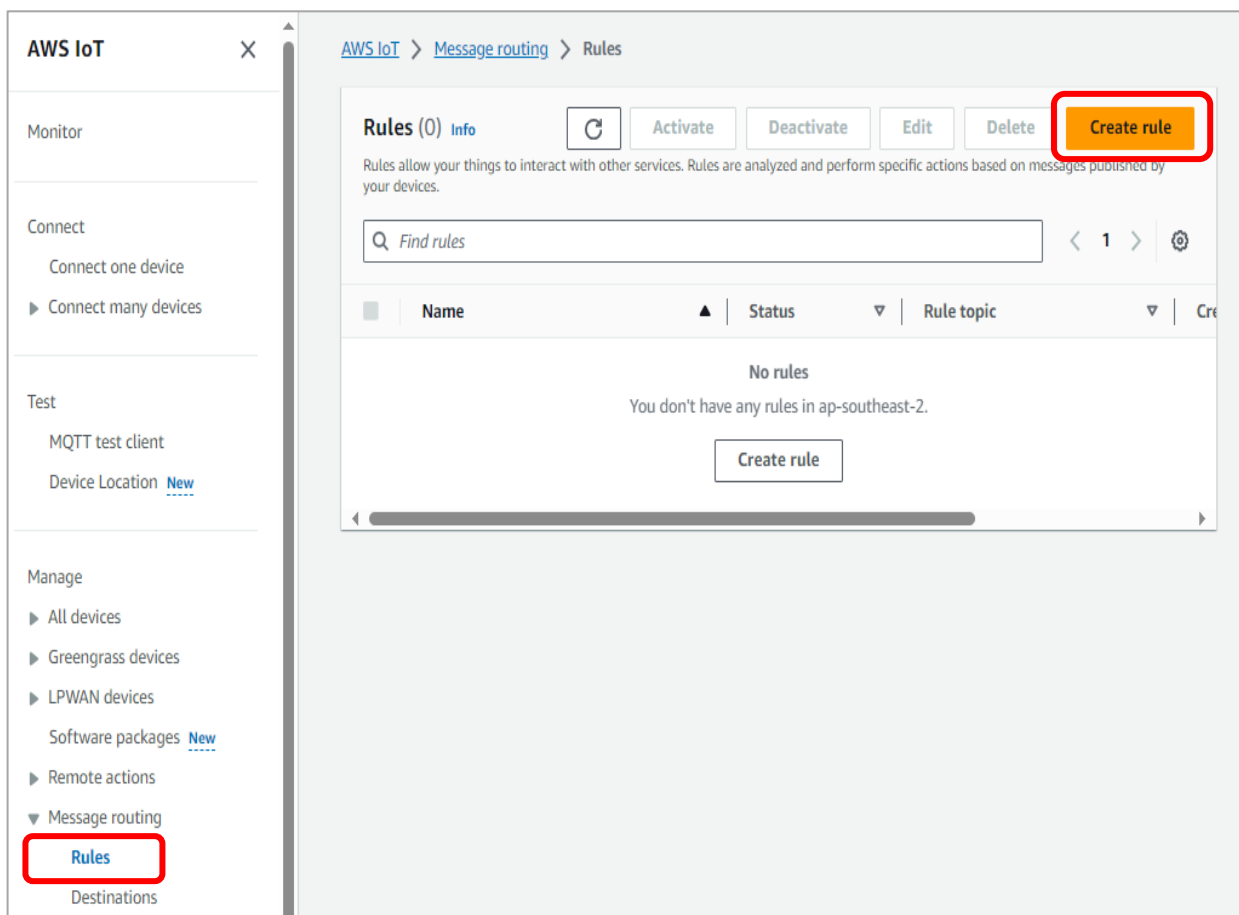


Figure 21: Create Rule

2. Enter a rule name and click **Next**.

DA16200 DA16600 Getting Started with AWS® IoT Core

AWS IoT > Message routing > Rules > Create rule

Step 1
Specify rule properties

Step 2
Configure SQL statement

Step 3
Attach rule actions

Step 4
Review and create

Specify rule properties [Info](#)

A rule resource contains a list of actions based on the MQTT topic stream.

Rule properties

Rule name

MyTestAct

Enter an alphanumeric string that can also contain underscore (_) characters, but no spaces.

Rule description - optional

Enter a description to provide additional details about the rule to others.

A description of your new rule

▼ Tags - optional

No tags are associated with the resource.

Add tag

You can add 1 more tag.

Cancel Next

Figure 22: Specify Rule Properties

3. Copy and paste the following SQL statement in the SQL statement box. See [Figure 23](#).

Note that the thing name is now **MyTestDoorLock**.

```
SELECT * FROM '$aws/things/Yourthingname/shadow/update'
WHERE state.reported.doorStateChange > 0 OR state.reported.temperature > 70 OR
state.reported.doorBell > 0
```

AWS IoT > Message routing > Rules > Create rule

Step 1
[Specify rule properties](#)

Step 2
Configure SQL statement

Step 3
Attach rule actions

Step 4
Review and create

Configure SQL statement [Info](#)

Add a simplified SQL syntax to filter messages received on an MQTT topic and push the data elsewhere.

SQL statement

SQL version
The version of the SQL rules engine to use when evaluating the rule.

2016-03-23

SQL statement
Enter a SQL statement using the following: SELECT <Attribute> FROM <Topic Filter> WHERE <Condition>. For example: SELECT

```
1 SELECT * FROM 'saws/things/MyTestDoorLock/shadow/update'
2 WHERE state.reported.doorStateChange > 0 OR state.reported
   .temperature > 70 OR state.reported.doorBell > 0
```

SQL Line 2, Column 106

Cancel Previous **Next**

Figure 23: Configure SQL Statement

4. Select **S3 bucket (Store messages in an Amazon S3 bucket)** from the dropdown menu. See [Figure 24](#).

DA16200 DA16600 Getting Started with AWS® IoT Core

AWS IoT > Message routing > Rules > Create rule

Step 1
[Specify rule properties](#)

Step 2
[Configure SQL statement](#)

Step 3
Attach rule actions

Step 4
[Review and create](#)

Attach rule actions Info

An action routes data to a specific AWS service.

SQL statement Back

```
SELECT * FROM '$aws/things/MyTestDoorLock/shadow/update'
WHERE state.reported.doorStateChange > 0 OR state.reported.
      temperature > 70 OR state.reported.doorBell > 0
```

Rule actions
Select one or more actions to happen when the above rule is matched by an inbound message. Actions define additional activities that occur when messages arrive, like storing them in a database, invoking cloud functions, or sending notifications. You can add up to 10 actions.

Action 1
▼ Choose an action Remove

Add rule action

Error action - optional
You can optionally set an action that will be executed when something goes wrong with processing your rule. If two rule actions in the same rule fail, the error action receives one message that contains both errors.

Add error action

Cancel Previous **Next**

Figure 24: Attach Rule Actions

- Click **Browse S3** and enter `${timestamp()}` for the Key. Then, click **Create new role**. See [Figure 25](#).

DA16200 DA16600 Getting Started with AWS® IoT Core

AWS IoT > Message routing > Rules > Create rule

Step 1
[Specify rule properties](#)

Step 2
[Configure SQL statement](#)

Step 3
Attach rule actions

Step 4
[Review and create](#)

Attach rule actions Info

An action routes data to a specific AWS service.

SQL statement Back

```
SELECT * FROM 'aws/things/HyTestDoorLock/shadow/update'
WHERE state.reported.doorStateChange > 0 OR state.reported.temperature > 70 OR state.reported.doorBell > 0
```

Rule actions

Select one or more actions to happen when the above rule is matched by an inbound message. Actions define additional activities that occur when messages arrive, like storing them in a database, invoking cloud functions, or sending notifications. You can add up to 10 actions.

Action 1

S3 bucket Remove

Store a message in an Amazon S3 bucket

Bucket name Info

S3 URL

Q s3:// X View 🔗 **Browse S3**

Key

The S3 key for this message.

s3key

Canned ACL

The Amazon S3 canned ACL that controls access to the object identified by the object key.

private

IAM role

Choose a role to grant AWS IoT access to your endpoint.

Choose an IAM role 🔄 View 🔗 **Create new role**

AWS IoT will automatically create a policy with a prefix of "aws-iot-rule" under your IAM role selection.

Add rule action

Error action - optional

You can optionally set an action that will be executed when something goes wrong with processing your rule. If two rule actions in the same rule fail, the error action receives one message that contains both errors.

Add error action

Cancel Previous Next

Figure 25: Attach Rule Actions (Continued)

6. Enter a IAM role name and click **Create**. See Figure 26.

Create role X

Role name

Log-MyAct-IAM-Role

Enter a unique role name that contains alphanumeric characters, hyphens, and underscores. A role name can't contain any spaces.

Cancel Create

Figure 26: Create IAM Role to Save Log Files

7. Review the entered information and click **Create**. See Figure 27.

DA16200 DA16600 Getting Started with AWS® IoT Core

[AWS IoT](#) > [Message routing](#) > [Rules](#) > Create rule

Step 1

[Specify rule properties](#)

Step 2

[Configure SQL statement](#)

Step 3

[Attach rule actions](#)

Step 4

Review and create

Review and create [Info](#)

Step 1: Rule properties

Edit

Rule properties

Name

MyTestAct

Description

-

Step 2: SQL statement

Edit

SQL statement

SQL version

2016-03-23

SQL query

SELECT * FROM '\$aws/things/MyTestDoorLock/shadow/update' WHERE state.reported.doorStateChange > 0 OR state.reported.temperature > 70 OR state.reported.doorBell > 0

Step 3: Rule actions

Edit

Actions

S3 bucket

Store a message in an Amazon S3 bucket

Bucket name

mytestdoorlock-log

Key

\${timestamp}

Canned ACL

private

IAM role

arn:aws:iam::649620604383:role/service-role/Log-MyAct-IAM-Role [\[?\]](#)

Error action

No error action

Cancel

Previous

Create

Figure 27: Review Rules

8. The created rules should appear in the list of policies. See [Figure 28](#).

DA16200 DA16600 Getting Started with AWS® IoT Core

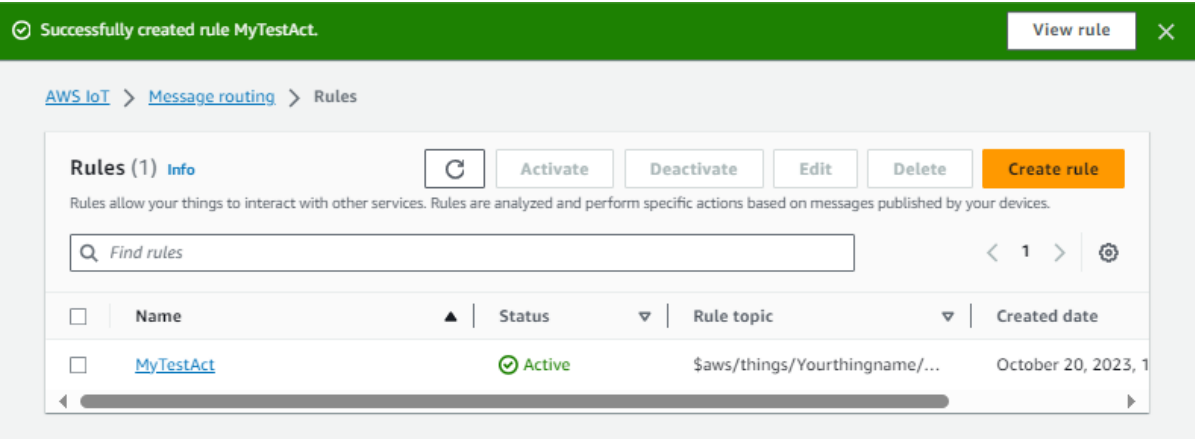


Figure 28: Created Act Rule

9. Now go to the IAM console and select **Role**, and check created roles. See Figure 29.

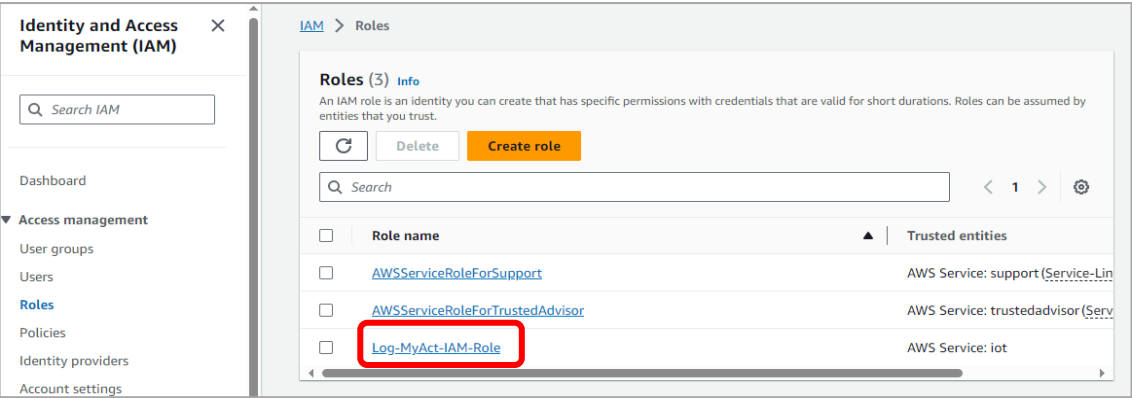


Figure 29: Created Act Role

10. Choose the created role name, and click **Attach policies**. See Figure 30.

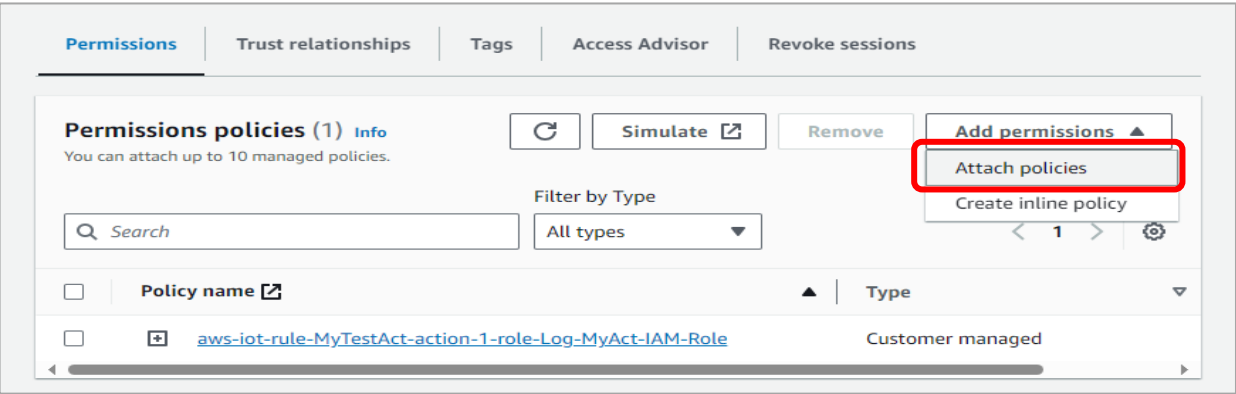
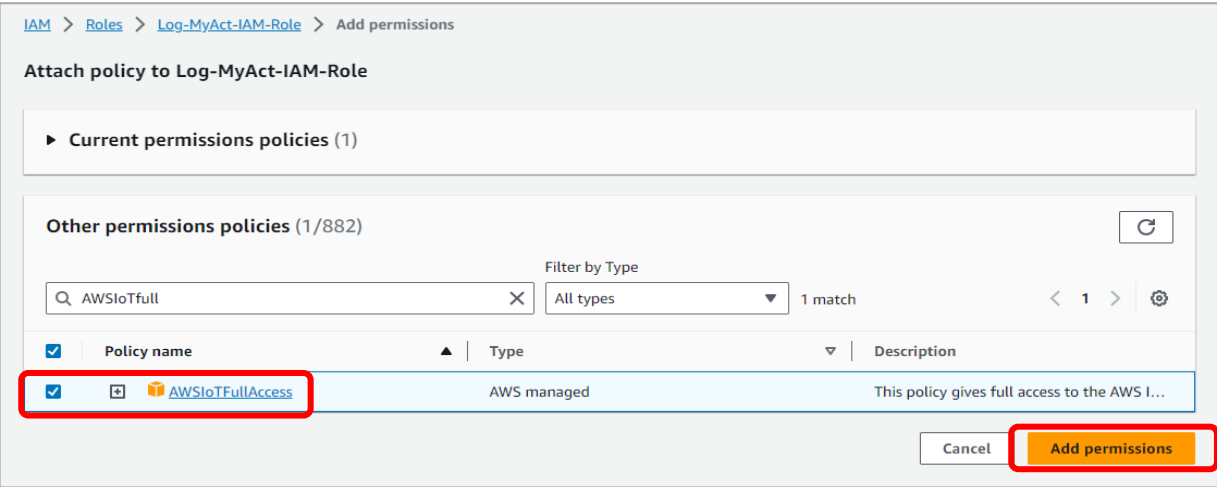


Figure 30: Attach Policy to Role

11. Search for the policy name of **AWSIoTFullAccess** and click **Add permissions**. Do the same thing for **AmazonS3FullAccess**. See Figure 31.

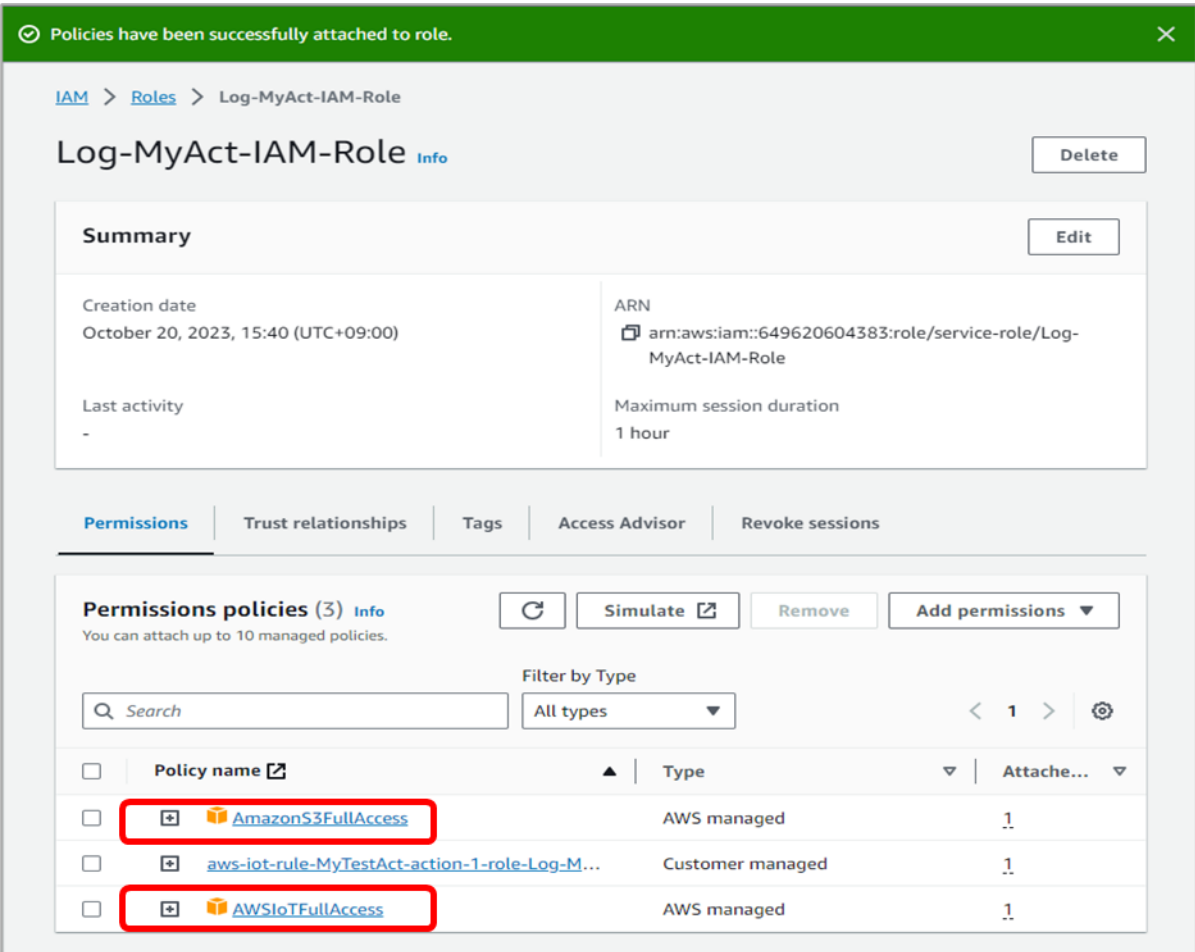
DA16200 DA16600 Getting Started with AWS® IoT Core



NOTE

AWSIoTFullAccess and AmazonS3FullAccess policies are not recommended for production.

12. Once the policies are added, the execution roles should look similar to [Figure 32](#).



DA16200 DA16600 Getting Started with AWS® IoT Core

3.1.3 Configure Amazon Cognito

Amazon Cognito provides authentication, authorization, and user management for web and mobile apps. Users can sign in directly with a username and password or through a third party such as Facebook, Amazon, or Google.

The two main components of Amazon Cognito are **user pools** and **identity pools**. User pools are directory of users that provide sign-up and sign-in options for app users. Identity pools provide AWS credentials to grant users access to other AWS services. Identity pools and user pools can be used separately or together. For more information, visit AWS Docs site (<https://docs.aws.amazon.com/cognito/latest/developerguide/what-is-amazon-cognito.html>).

3.1.3.1 Create User Pools

1. Go to the Amazon Cognito console and click **Create user pool** to create a user pool. See [Figure 33](#).

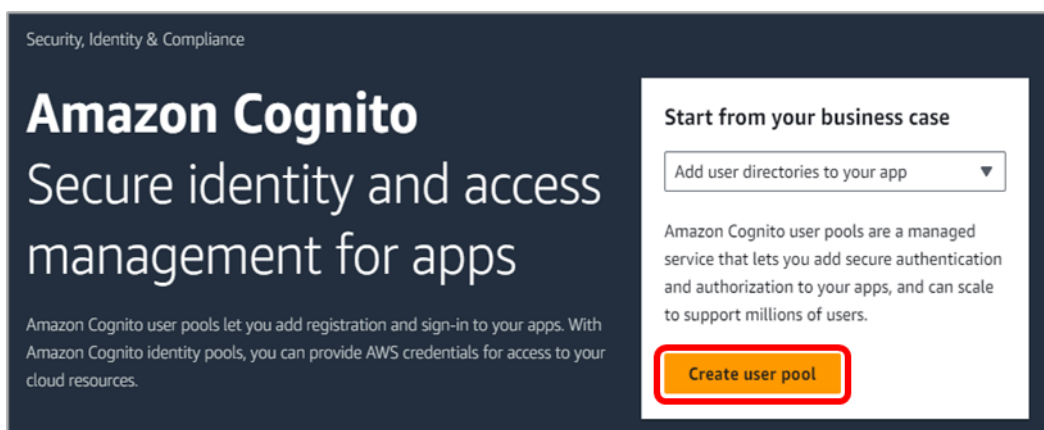


Figure 33: Create User Pool

2. In the **Configure sign-in experience** page, choose Email, and click **Next**. See [Figure 34](#).

DA16200 DA16600 Getting Started with AWS® IoT Core

Amazon Cognito > User pools > Create user pool

Step 1
Configure sign-in experience

Step 2
Configure security requirements

Step 3
Configure sign-up experience

Step 4
Configure message delivery

Step 5
Integrate your app

Step 6
Review and create

Configure sign-in experience [Info](#)

Your app users can sign in to your user pool with a user name and password, or sign in with a third-party identity provider.

Authentication providers

Configure the providers that are available to users when they sign in.

Provider types
Choose whether users will sign in to your Cognito user pool, a federated identity provider, or both. Amazon Cognito has different pricing for federated users and user pool users. [Learn more about pricing](#)

☒ **Cognito user pool**
Users can sign in using their email address, phone number, or user name. User attributes, group memberships, and security settings will be stored and configured in your user pool.

☐ **Federated identity providers**
Users can sign in using credentials from social identity providers like Facebook, Google, Amazon, and Apple; or using credentials from external directories through SAML or Open ID Connect. You can manage user attribute mappings and security for federated users in your user pool.

Cognito user pool sign-in options [Info](#)

Choose the attributes in your user pool that are used to sign in. If you select only one attribute, or you select a user name and at least one other attribute, your user can sign in with all of the selected options. If you select only phone number and email, your user will be prompted to select one of the two sign-in options when they sign up.

☐ User name

☒ Email

☐ Phone number

⚠ Cognito user pool sign-in options can't be changed after the user pool has been created.

Cancel **Next**

Figure 34: Configure Sign-in Options

3. Choose Cognito defaults as password policy mode. Then, select No MFA and Email only, and click **Next**. See [Figure 35](#).

DA16200 DA16600 Getting Started with AWS® IoT Core

Amazon Cognito > User pools > Create user pool

Step 1
[Configure sign-in experience](#)

Step 2
Configure security requirements

Step 3
[Configure sign-up experience](#)

Step 4
[Configure message delivery](#)

Step 5
[Integrate your app](#)

Step 6
[Review and create](#)

Configure security requirements [Info](#)

Set up a strong password requirement in addition to multi-factor authentication to protect your app users from accidentally compromising their credentials.

Password policy [Info](#)

Create a password policy to define the length and complexity of the passwords your users can set.

Password policy mode [Info](#)

☒ **Cognito defaults**
Use default password requirements.

☐ Custom
Use password requirements that you define.

Password minimum length
8 character(s)

Password requirements
Contains at least 1 number
Contains at least 1 special character
Contains at least 1 uppercase letter
Contains at least 1 lowercase letter

Temporary passwords set by administrators expire in
7 day(s)

Multi-factor authentication

Configure secure access to your app by enforcing multi-factor authentication (MFA) during the user sign-in process. MFA settings are applied to all app clients.

MFA enforcement [Info](#)

☐ Require MFA - Recommended
Users must provide an additional authentication factor when signing in.

☐ Optional MFA
Users can sign in with a single authentication factor, and can choose to add additional authentication factors.

☒ **No MFA**
Users can only sign in with a single authentication factor. This is the least secure option.

User account recovery

Configure how users will recover their account when they forget their password. Recipient message and data rates apply.

Self-service account recovery [Info](#)

☒ **Enable self-service account recovery - Recommended**
Allow forgot-password operations in your user pool. In the hosted UI sign-in page, a "Forgot your password?" link is displayed. When this feature is not enabled, administrators reset passwords with the Cognito API.

Delivery method for user account recovery messages [Info](#)

Select how your user pool will deliver messages when users request an account recovery code. SMS messages are charged separately by Amazon SNS. Email messages are charged separately by Amazon SES. [Learn more about pricing.](#)

☒ **Email only**

☐ SMS only

☐ Email if available, otherwise SMS

☐ SMS if available, otherwise email

☐ SMS if available, otherwise email, and allow a user to reset their password via SMS if they are also using it for MFA

Cancel Previous **Next**

Figure 35: Configure Security Requirements

- Choose items as shown in [Figure 36](#).

DA16200 DA16600 Getting Started with AWS® IoT Core

Amazon Cognito > User pools > Create user pool

Step 1
[Configure sign-in experience](#)

Step 2
[Configure security requirements](#)

Step 3
Configure sign-up experience

Step 4
[Configure message delivery](#)

Step 5
[Integrate your app](#)

Step 6
[Review and create](#)

Configure sign-up experience [Info](#)

Determine how new users will verify their identities when signing up and which attributes should be required or optional during the user sign-up flow.

Self-service sign-up [Info](#)

Choose whether new users of your app can register for an account themselves.

Self-registration [Info](#)

☒ **Enable self-registration**
Display a "Sign up" link on the sign-in page in the hosted UI, and allow the use of public APIs to create new user accounts. When this feature is not enabled, federation and administrative API operations create user profiles.

! If you activate user sign-up in your user pool, anyone on the internet can sign up for an account and sign in to your apps. Don't enable self-registration in your user pool until you want to open your app to public sign-up. [Learn more](#)

Attribute verification and user account confirmation

Choose between Cognito-assisted and self-managed user attribute verification and account confirmation. Only verified attributes can be used for sign-in, account recovery, and MFA. A user account must be confirmed either by attribute verification, or user pool administrator confirmation, before a user is allowed to sign in.

Cognito-assisted verification and confirmation [Info](#)

☒ **Allow Cognito to automatically send messages to verify and confirm - Recommended**
Cognito sends a verification message with a code that the user must enter. For new users, this will verify the attribute and confirm their account. When this feature is not enabled, administrative API operations and Lambda triggers verify and confirm users.

Attributes to verify [Info](#)

Choose the user contact attribute that Cognito will send a verification message to. Recipient message and data rates apply when you use SMS.

☐ **Send SMS message, verify phone number**
Verify with SMS to allow users to use their phone number for sign-in, MFA, and account recovery. SMS messages are charged separately by Amazon SNS.

☒ **Send email message, verify email address**
Verify with email to allow users to use their email address for sign-in, MFA, and account recovery. Email messages are charged separately by Amazon SES.

☐ **Send SMS message if phone number is available, otherwise send email message**
You must build custom code when you want to verify both email and phone numbers at user account creation.

Verifying attribute changes [Info](#)

☒ **Keep original attribute value active when an update is pending - Recommended**
When you update the value of an email or phone number attribute, your user must verify the new value. Until they verify the new value, they can receive messages and sign in with the original value. If you don't turn on this feature, your user can't sign in with that attribute before they verify the new value.

Active attribute values when an update is pending [Info](#)

Choose the attributes that you want to keep active when an update to their value is pending. Your users can receive messages and sign in with the original attribute value until they verify the new value.

☐ Phone number

☒ **Email address**

☐ Phone number and email address

Required attributes [Info](#)

Choose the attributes that are required when a new user is created. Cognito assigns all users a set of standard attributes based on the OpenID Connect (OIDC) standard.

Required attributes based on previous selections
email

Additional required attributes

! Required attributes can't be changed once this user pool has been created.

Custom attributes - optional

Personalize the sign-up experience by adding up to 50 custom attributes. Custom attribute names can't be changed after a user pool has been created.

Cancel Previous **Next**

Figure 36: Configure Security Requirements (Continued)

5. Choose **Send email with Cognito**. See [Figure 37](#).

DA16200 DA16600 Getting Started with AWS® IoT Core

The screenshot shows the 'Configure message delivery' step in the AWS IAM console. The breadcrumb trail is 'Amazon Cognito > User pools > Create user pool'. The left sidebar shows a progress bar with steps 1 through 6. Step 4, 'Configure message delivery', is the current step. The main content area is titled 'Configure message delivery' with an 'Info' icon. Below the title is a note: 'Amazon Cognito uses Amazon SES and Amazon SNS to send email and SMS messages to your app users. Messages may incur additional SES and SNS costs.' The 'Email' section is active, with the instruction 'Configure how your user pool sends email messages to users.' Under 'Email provider', there are two radio buttons: 'Send email with Amazon SES - Recommended' (unselected) and 'Send email with Cognito' (selected). The 'Send email with Cognito' option has a description: 'Use Cognito's default email address as a temporary start for development. You can use it to send up to 50 emails a day.' Below this, a note states: 'You must have configured a verified sender with Amazon SES to use the SES feature. Learn more'. The 'SES Region' is set to 'Asia Pacific (Seoul)'. The 'FROM email address' is set to 'no-reply@verificationemail.com' with a refresh button. The 'REPLY-TO email address - optional' field is empty with a placeholder 'Enter an email address'. At the bottom right are 'Cancel', 'Previous', and 'Next' buttons.

Amazon Cognito > User pools > Create user pool

Step 1
[Configure sign-in experience](#)

Step 2
[Configure security requirements](#)

Step 3
[Configure sign-up experience](#)

Step 4
Configure message delivery

Step 5
[Integrate your app](#)

Step 6
[Review and create](#)

Configure message delivery [Info](#)

Amazon Cognito uses Amazon SES and Amazon SNS to send email and SMS messages to your app users. Messages may incur additional SES and SNS costs.

Email

Configure how your user pool sends email messages to users.

Email provider [Info](#)

☐ Send email with Amazon SES - Recommended
Send emails using an Amazon SES verified identity in your account. We recommend this option for higher email volume and production workloads.

☒ Send email with Cognito
Use Cognito's default email address as a temporary start for development. You can use it to send up to 50 emails a day.

You must have configured a verified sender with Amazon SES to use the SES feature. [Learn more](#)

SES Region [Info](#)
Asia Pacific (Seoul)

FROM email address [Info](#)
By default "no-reply@verificationemail.com" will be used. You can also choose a different email address that you have previously verified with Amazon SES.

no-reply@verificationemail.com

REPLY-TO email address - optional [Info](#)
If you set an invalid reply-to address, sending restrictions may be imposed on your account.

Enter an email address

Cancel Previous Next

Figure 37: Configure Message Delivery

6. In the **Integrate your app** page, enter required items as shown in [Figure 38](#) and click **Next**.

DA16200 DA16600 Getting Started with AWS® IoT Core

Amazon Cognito > User pools > Create user pool

Step 1
Configure sign-in experience

Step 2
Configure security requirements

Step 3
Configure sign-up experience

Step 4
Configure message delivery

Step 5
Integrate your app

Step 6
Review and create

Integrate your app

Set up app integration for your user pool with Cognito's built-in authentication and authorization flows.

User pool name

Create a friendly name for your user pool.

User pool name

MyUserPool_DoorLock

User pool names are limited to 128 characters or less. Names may only contain alphanumeric characters, spaces, and the following special characters: + = . , @ -

Your user pool name can't be changed once this user pool is created.

Hosted authentication pages

Choose whether to use Cognito's Hosted UI and OAuth 2.0 server for user sign-up and sign-in flows.

Use the Cognito Hosted UI

Build hosted sign-up, sign-in, and OAuth 2.0 service endpoints in Amazon Cognito. When this feature is not enabled, use Cognito API operations to perform sign-up and sign-in.

Initial app client

Configure an app client. App clients are single-app platforms in your user pool that have permissions to call unauthenticated API operations. A user pool can have multiple app clients.

App type

Select an app type and we will automatically populate common default settings. You can add additional app clients after the user pool is created.

Public client

A native, browser or mobile-device app. Cognito API requests are made from user systems that are not trusted with a client secret.

Confidential client

A server-side application that can securely store a client secret. Cognito API requests are made from a central server.

Other

A custom app. Choose your own grant, auth flow, and client-secret settings.

App client name

Enter a friendly name for your app client.

MyAppClient

App client names are limited to 128 characters or less. Names may only contain alphanumeric characters, spaces, and the following special characters: + = . , @ -

Client secret

Choose whether your app client will have a client secret. Client secrets are used by the server-side component of an app to authorize API requests and prevent a third party from impersonating your client.

Generate a client secret

Don't generate a client secret

You cannot change or remove a client secret after you allow Amazon Cognito to generate it for your app client.

Advanced app client settings

We have populated suggested authentication flows, OAuth 2.0 grant types, and OIDC scopes based on the selections you made earlier.

Attribute read and write permissions

Choose the standard and custom attributes this app can read and write. Required attributes are locked as writable. We recommend that you set immutable custom attributes as writable to allow the app client to set initial values during sign-up.

Tags (0) - optional

You can add tags to your user pool for cost management and access control.

No tags associated with the resource.

Add new tag

You can add up to 50 tags.

Cancel

Previous

Next

Figure 38: Integrate App Client

7. In the **Review and create** page, review the entered information, and click **Create user pool**. Then, the created user pool should appear in the list. See [Figure 39](#).

User Manual

Revision 1.5

Jan. 26, 2024

CFR0012

36 of 94

© 2024 Renesas Electronics

DA16200 DA16600 Getting Started with AWS® IoT Core

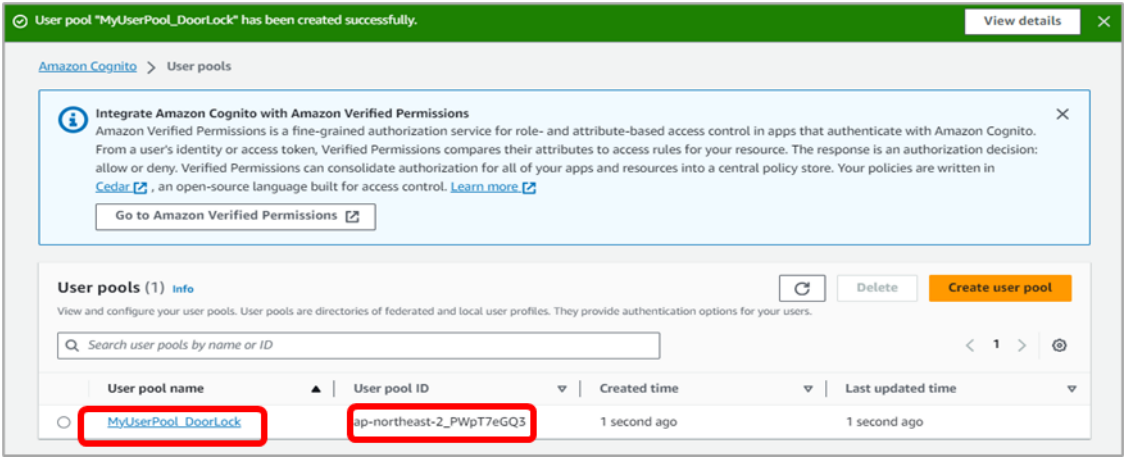


Figure 39: Created User Pool

3.1.3.2 Create Identity Pools

1. Go to the Amazon Cognito console. Choose **Identity pools** and click **Create identity pool**. See Figure 40.

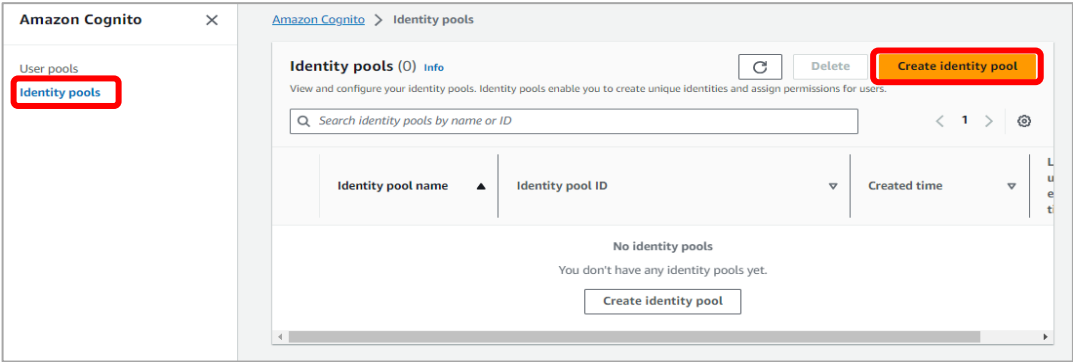


Figure 40: Create Identity Pool

2. In the **Configure identity pool trust** page, choose **Guest access**. See Figure 41.

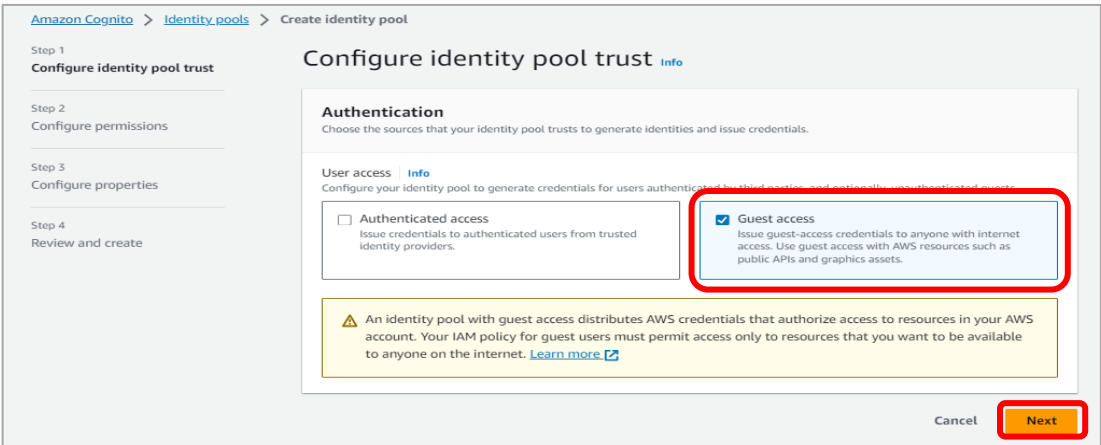


Figure 41: Create Identity Pool Trust

3. In the **Configure permissions** page, select **Create a new IAM role** and enter a **IAM role name**.

DA16200 DA16600 Getting Started with AWS® IoT Core

Amazon Cognito > Identity pools > Create identity pool

Step 1
[Configure identity pool trust](#)

Step 2
Configure permissions

Step 3
[Configure properties](#)

Step 4
[Review and create](#)

Configure permissions Info

Each Amazon Cognito identity pool can have a default AWS Identity and Access Management (IAM) role for authenticated and unauthenticated identities. If you don't change the default authenticated role for an identity provider, your users will receive credentials for the authenticated identities role.

Guest role Info
Configure the default IAM role that your anonymous guest users will assume through your identity pool.

IAM role
Choose a role to use as the default for your Amazon Cognito users to assume one for you.

☒ **Create a new IAM role**
Create an IAM role with basic permissions and a trust relationship with your identity pool.

☐ **Use an existing IAM role**
Choose a role in your account that you have configured to trust cognito-identity.amazonaws.com.

IAM role name
Enter a name for your new IAM role.
Cognito_MyTestDoorLockG
Role names may be up to 64 characters long and can use alphanumeric characters, as well as the following special characters: +, -, @, _.

You're creating an IAM role with initial minimum permissions and a trust relationship with your identity pool.
After you create your identity pool, add permissions in the IAM console.

View policy document
View the policy document that Amazon Cognito has generated.

Cancel Previous **Next**

Figure 42: Configure Permissions

4. Enter an Identity pool name and click **Next**.

Amazon Cognito > Identity pools > Create identity pool

Step 1
[Configure identity pool trust](#)

Step 2
[Configure permissions](#)

Step 3
Configure properties

Step 4
[Review and create](#)

Configure properties Info

Identity pool name
Create a friendly name for your identity pool.

Name
MyIdentityPool_DoorLockG
Identity pool names are limited to 128 characters or less. Names may only contain alphanumeric characters, spaces, and the following special characters: +, -, @, _.

Basic (classic) authentication Info
Activate the classic authentication flow if your app relies on separate API requests to retrieve an identity token, and then to assume a role using that token. When you activate the classic flow, you can still use the recommended enhanced flow.

Basic authentication
☐ **Activate basic flow**

Tags (0) - optional Info
Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this identity pool.

No tags associated with the resource.

Add new tag
You can add up to 50 tags.

Cancel Previous **Next**

Figure 43: Configure Properties

5. Review the selected items and click **Create identity pool**. Then, the created identity pools appear in the list. See [Figure 44](#).

DA16200 DA16600 Getting Started with AWS® IoT Core

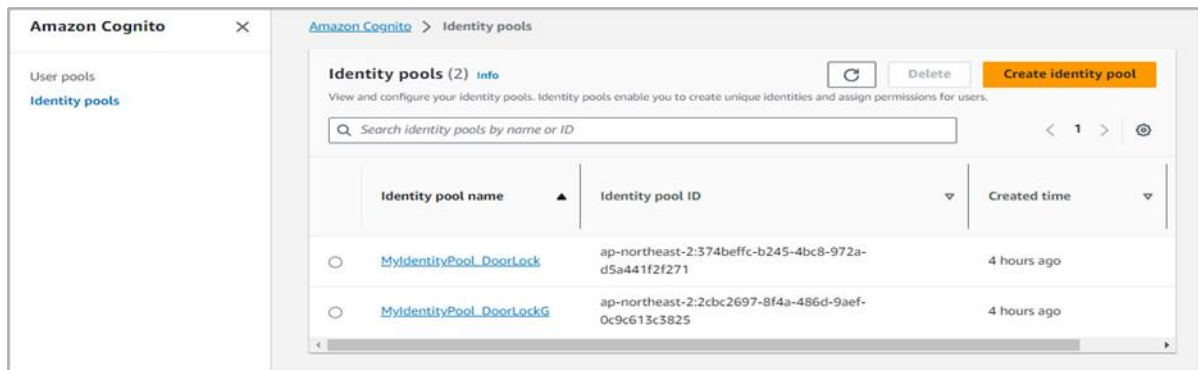


Figure 44: Created Identity Pools

3.1.4 Set Up AWS IAM

While creating an identity pool, users should update the IAM roles that the users assume. When a user logs in to the app, Amazon Cognito generates temporary AWS credentials for the user. These temporary credentials are associated with a specific IAM role. The IAM role lets users to define a set of permissions to access AWS resources. For more information, visit <https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html>.

- The roles in Cognito_MyTestDoorlockG are created automatically when the federation identity is created via Cognito Identity Pool
- The device only needs an unauthorized role

1. Go to the IAM console, and choose **Cognito_MyTestDoorLockG**. See Figure 45.

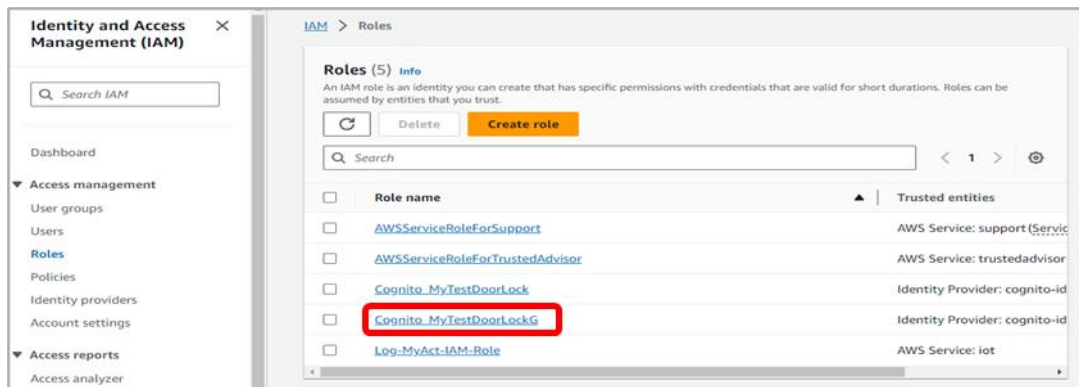


Figure 45: IAM Role

2. Click **Attach policies**. See Figure 46.

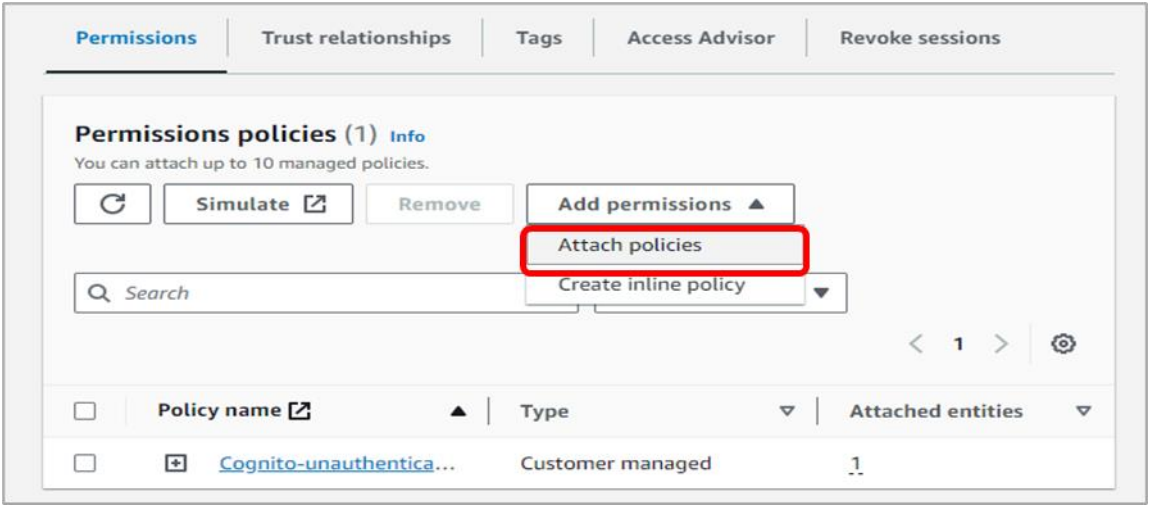


Figure 46: Attach Policies

3. Search for the policy name of **AWSIoTFullAccess** and click **Attach permissions**. See [Figure 47](#).

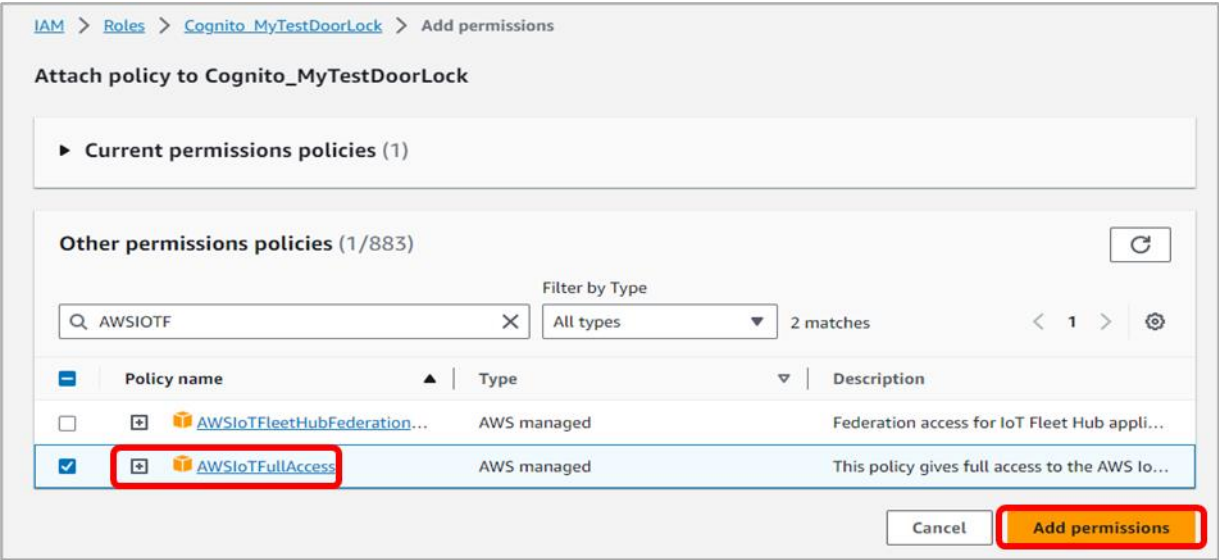


Figure 47. AWSIoTFullAccess Policy

NOTE
AWSIoTFullAccess policy is not recommended for production.

4. Search for the policy name of **AmazonS3FullAccess** and click **Attach permissions**. See [Figure 48](#).



Figure 48: AmazonS3FullAccess Policy

DA16200 DA16600 Getting Started with AWS® IoT Core

NOTE

AmazonS3FullAccess policy is not recommended for production.

5. The attached policies appear in the list. See [Figure 49](#).

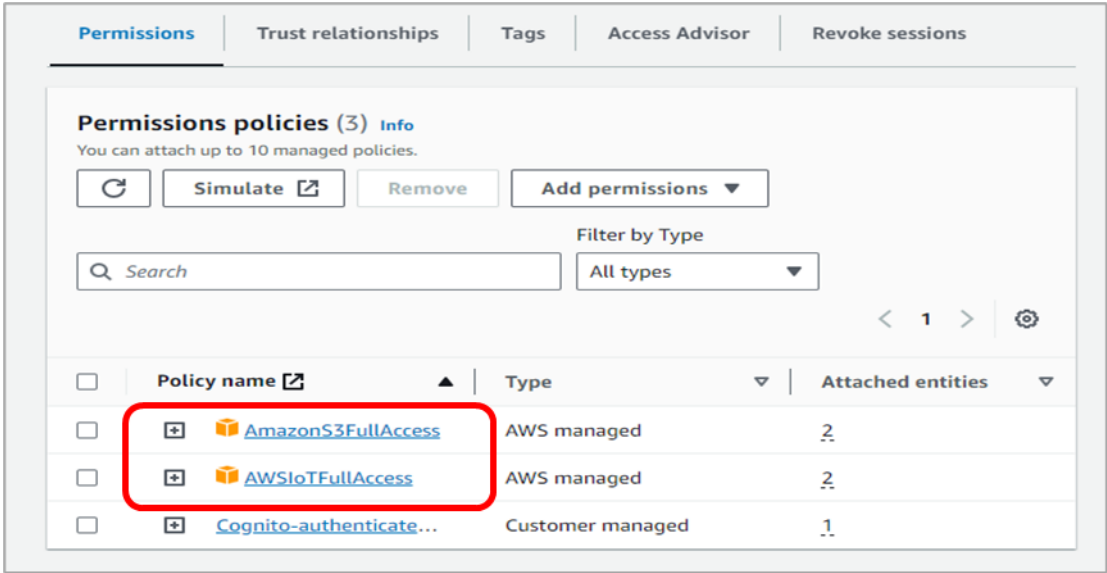


Figure 49: Attached Policies

3.1.5 Create Amazon S3 Bucket

Every object in Amazon S3 is stored in a bucket. Before storing data in Amazon S3, users need to create a bucket. To create S3 bucket, complete the following steps.

1. In the Amazon S3 console, choose **Buckets** in the left navigation pane, and click **Create bucket**.
2. In the **Create bucket** page, type a bucket name in the Bucket name field.
3. For **Region**, choose the AWS region where a user wants the bucket to reside, and click **Create bucket**.

When Amazon S3 successfully creates the bucket, the console displays an empty bucket in the **Buckets** pane.

DA16200 DA16600 Getting Started with AWS® IoT Core

4 Door Lock Reference Application

Door lock reference application is available in the official website.

NOTE

Go to the Renesas website (<https://www.renesas.com/us/en/products/wireless-connectivity/wi-fi/low-power-wi-fi>) and scroll down to the Software Downloads section. Find "AWS IoT Reference" or type it in the search box, and then select the reference package and download.

4.1 Reference Application in DA16200/DA16600

The following components shown in [Figure 50](#) are required to run the application in DA16200/DA16600 via an Internet connection and AWS IoT server.

- AWS IoT reference application package
- DA16200/DA1660 EVB
- Router: Connection to internet
- Mobile device: Android/iOS application
- AWS account



Figure 50: Architecture of AWS IoT

Install the mobile application by searching for **DA16200** or **DA16600** on the Google Play Store or Apple App Store on the mobile devices.

Provisioning is required for connection between DA16200/DA16600 and Router before connecting DA16200/DA16600 with AWS IoT hub. The provisioning can be done with the Renesas Wi-Fi Provisioning app on either an Android or iOS device. See Ref. [4] for details on how to install and provision the mobile app. Once provisioning is completed, select AWS IoT to open AWS application on mobile device.

4.1.1 Open Door

[Figure 51](#) shows message flows of opening the door.

DA16200 DA16600 Getting Started with AWS® IoT Core

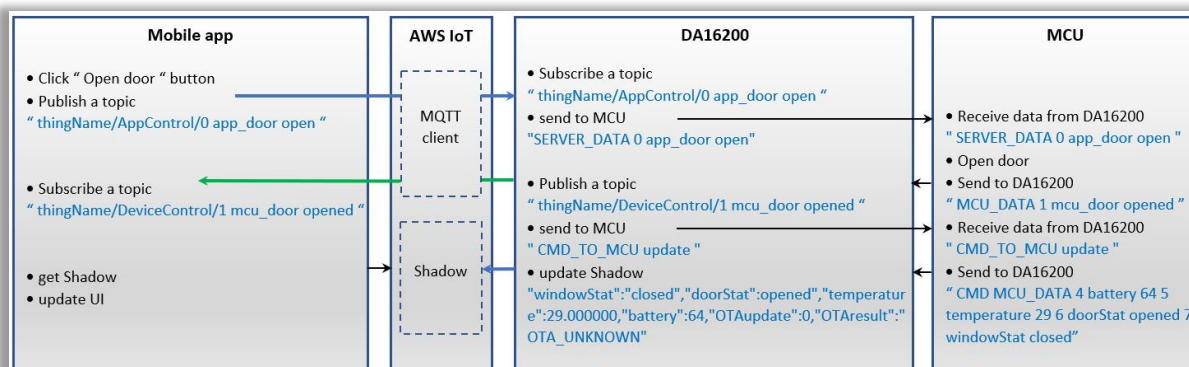


Figure 51: Message Flows of Opening Door

The operation of **opening door** in Android App is shown in Figure 53.

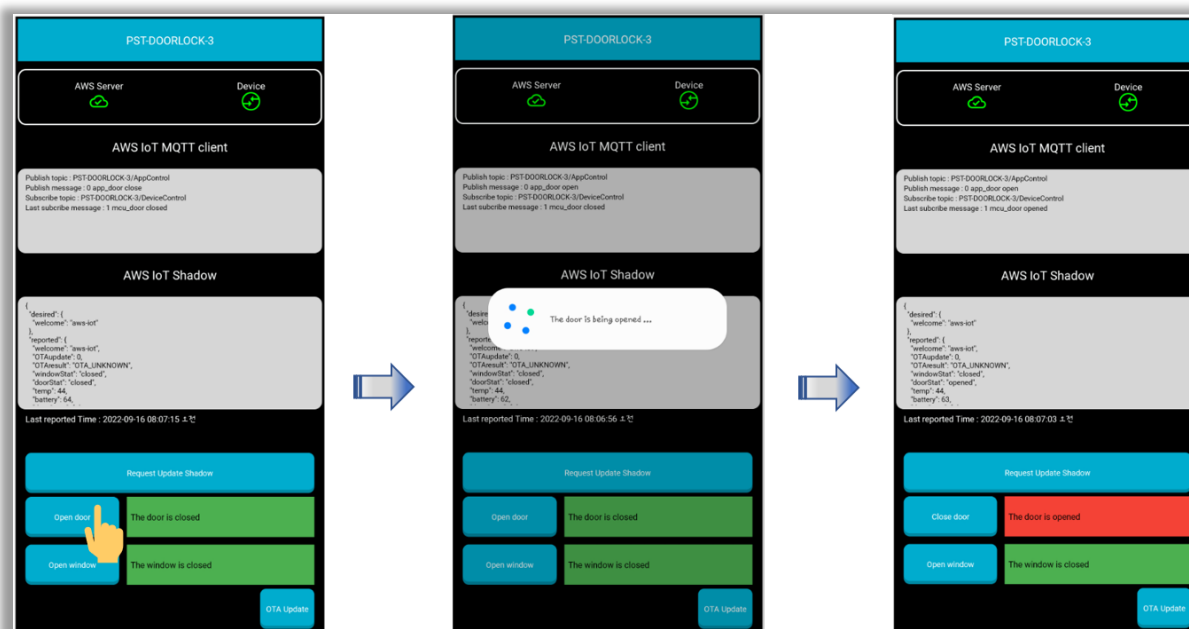


Figure 52: Open Dooring on Mobile App

DA16200 DA16600 Getting Started with AWS® IoT Core



Figure 53: Shadow State When Door is Open

When the operation of opening door is completed, the console logs of DA16200 are displayed as below.

```

Count : 0, cmdNum = 4
mqtttype = 1
index(=3) matched
data type(shadow) = 0
call update sensor(need to be set variable): battery = 63

Count : 1, cmdNum = 5
mqtttype = 1
index(=2) matched
data type(shadow) = 2
call update sensor(need to be set variable): temperature = 28.000000

Count : 2, cmdNum = 6
mqtttype = 1
index(=1) matched
data type(shadow) = 1
call update sensor(need to be set variable): doorStat = opened

Count : 3, cmdNum = 7
mqtttype = 1
index(=0) matched
data type(shadow) = 1
call update sensor(need to be set variable): windowStat = closed

release response

*****
publish (shadow sensor update) OK - payload:
{"state":{"reported":{"windowStat":"closed","doorStat":"opened","temperature":28.000000,"battery":63,"OTAupdate":0,"OTAresult":"OTA_UNKNOWN"}},"clientToken":"PST-DOORLOCK-3-0"}

```

4.1.2 Close Door

Figure 54 shows message flows of closing door.

DA16200 DA16600 Getting Started with AWS® IoT Core

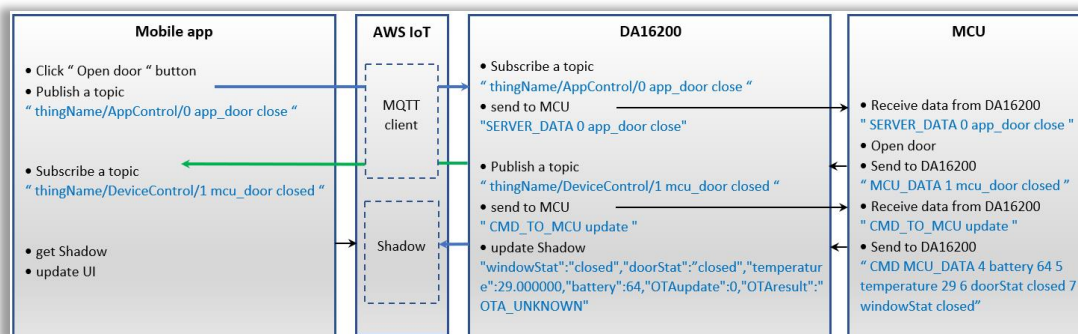


Figure 54: Message Flows of Closing Door

The operation of **closing door** in Android App is shown in Figure 55.

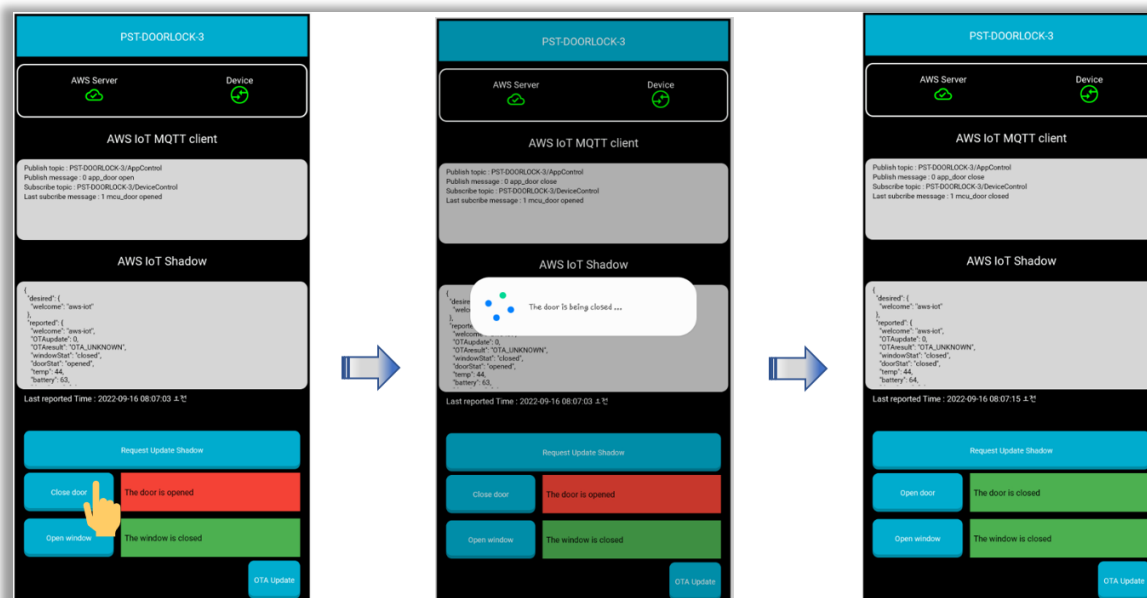


Figure 55: Closing Door on Mobile App

Figure 56 shows the state of Shadow on the AWS IoT Hub when the operation for closing door is completed.

Device Shadow state

```
{
  "state": {
    "desired": {
      "welcome": "aws-iot"
    },
    "reported": {
      "welcome": "aws-iot",
      "OTAupdate": 0,
      "OTAresult": "OTA_UNKNOWN",
      "windowStat": "closed",
      "doorStat": "closed",
      "temp": 44,
      "battery": 76,
      "doorState": false,
      "openMethod": "app",
      "doorStateChange": 1,
      "DoorOpenMode": 0,
      "temperature": 41,
    }
  }
}
```

Figure 56: Shadow State when Door is Closed

When the operation of closing door is completed, the console logs of the DA16200/DA16600 are displayed as below.

```
Count : 0, cmdNum = 4
mqtttype = 1
index(=3) matched
data type(shadow) = 0
call update sensor(need to be set variable): battery = 76

Count : 1, cmdNum = 5
mqtttype = 1
index(=2) matched
data type(shadow) = 2
call update sensor(need to be set variable): temperature = 41.000000

Count : 2, cmdNum = 6
mqtttype = 1
index(=1) matched
data type(shadow) = 1
call update sensor(need to be set variable): doorStat = closed

Count : 3, cmdNum = 7
mqtttype = 1
index(=0) matched
data type(shadow) = 1
call update sensor(need to be set variable): windowStat = closed

release response

*****
publish (shadow sensor update) OK - payload:
{"state":{"reported":{"windowStat":"closed","doorStat":"closed","temperature":41.000000,"battery":76,"OTAupdate":0,"OTAresult":"OTA_UNKNOWN"}}, "clientToken":"PST-DOORLOCK-3-0"}
```

DA16200 DA16600 Getting Started with AWS® IoT Core

4.2 Reference Application in Host MCU

Application in host MCU can control DA16200/DA16600 and connection between the host MCU and mobile phone through AWS IoT server using AT commands. Figure 57 shows the AWS IoT using firmware images for AT commands and host MCU.

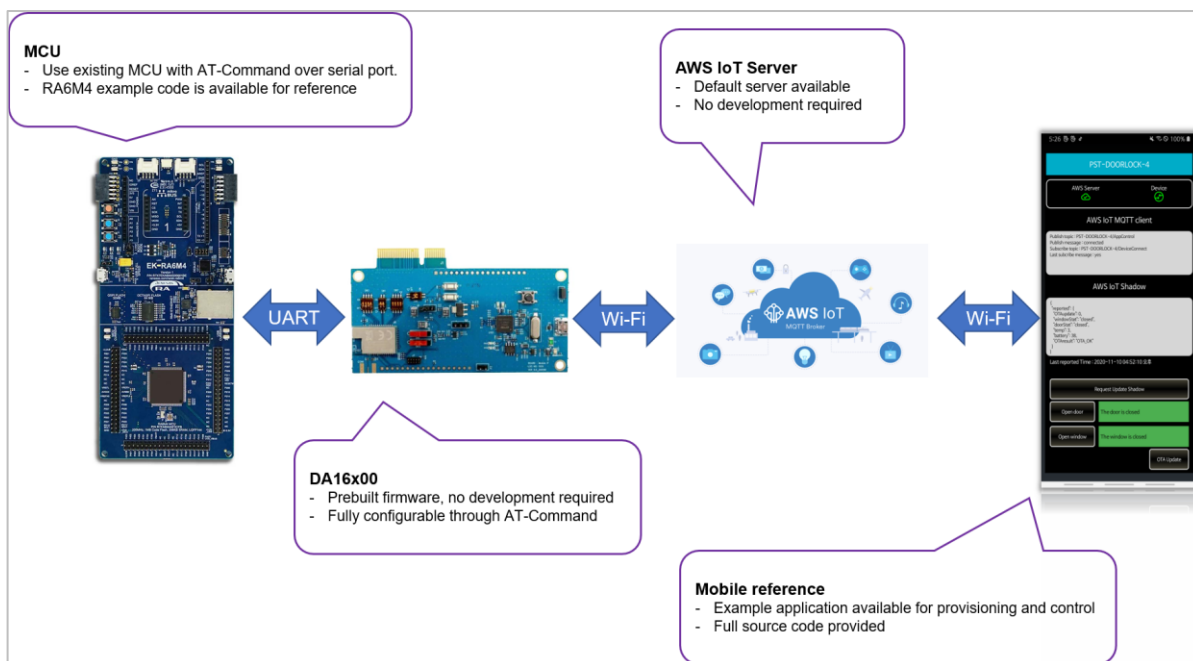


Figure 57: AWS IoT Using Firmware Images for AT Commands and Host MCU

4.2.1 Download Package for Door Lock Reference Application in Host MCU

A firmware image for AT command and application in MCU are available in the official Renesas website (<https://www.renesas.com/us/en/products/wireless-connectivity/wi-fi/low-power-wi-fi>).

The contents of the package are as follows:

- \DA16200 or \DA16600
 - Firmware images for the DA16200/DA16600 Wi-Fi devices
 - Tera Term script for downloading the firmware images to a DA16200/DA16600 Wi-Fi device
- \DA16200\Script (\DA16600\Script)
 - Tera Term script which demonstrates how to use AT commands for AWS IoT using a personal computer and the DA16200/DA16600
 - Getting Started with AT commands for AWS IoT
 - Introduces the DA16200/DA16600 AT commands for AWS IoT and describes how to set up the development environment and test the examples.
 - Describes how to connect an external host to the DA16200/DA16600 EVK for using the AT commands for AWS IoT.
 - Describes the AT commands for AWS IoT command list.
- \MCU
 - Sample project based on the RA6M4 development environment which demonstrates how to use AT commands for AWS IoT

DA16200 DA16600 Getting Started with AWS® IoT Core

4.2.2 Hardware Connections between DA16200/DA16600 and Host MCU

The hardware components listed in [Figure 58](#) are required to run door lock reference application using AT commands and host MCU.

- DA16200/DA16600 EVK
- EK-RA6M4 board
- Windows laptop or personal computer

In addition, hardware connections listed below are required for each operation.

- UART0: Programming firmware images and monitoring logs from DA16200/DA16600
- UART1 or UART2: AT command interface between MCU and DA16200/DA16600
- GPIO from the MCU to the DA16200/DA16600 to wake up the DA16200/DA16600 from DPM low power mode (DPM LPM)
- GPIO from the DA16200/DA16600 to host MCU to wake up the MCU in sleep mode

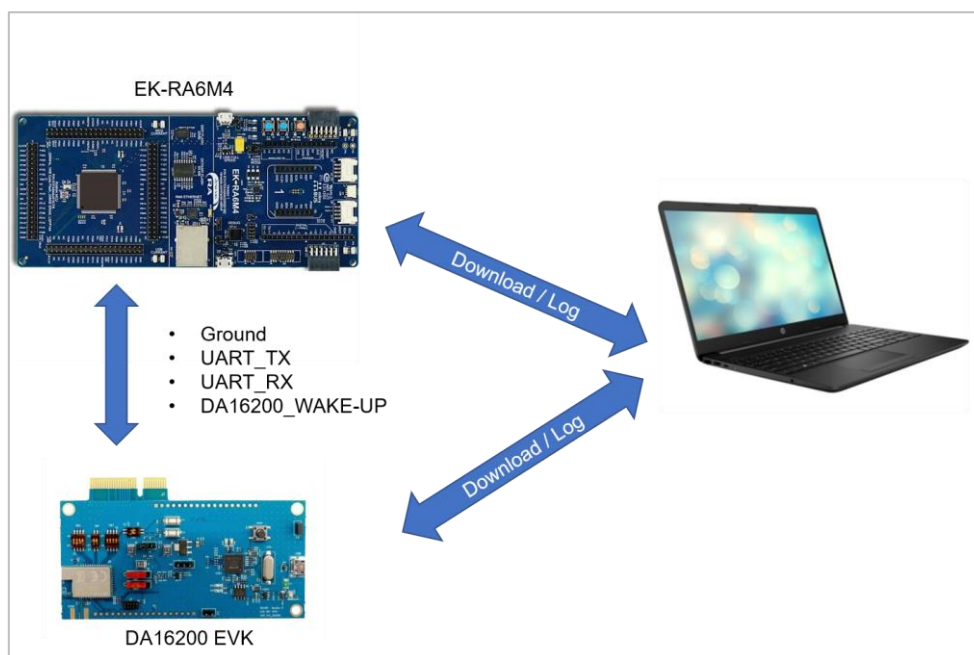


Figure 58: Hardware Configuration

The pin connections between the DA16200/DA16600 EVK and the EK-RA6M4 board are shown in [Table 1](#).

Table 1: Pin Connection

Function	DA16200 EVK		DA16600 EVK		EK-RA6M4 Board	
	Pin Number	Pin Name	Pin Number	Pin Name	Pin Number	Pin Name
Ground	J3.18	GND	J2.12	GND	J24-7	GND
UART_TX	J4.11	TX1/GPIOA_4	J2.2	TX2/GPIOC_6	J23-2	D1 / TXD
UART_RX	J4.12	RX1/GPIOA_5	J2.4	RX2/GPIOC_7	J23-1	D0 / RXD
DA16200_WAKE_UP	J3.11	RTC_WAKE_UP2	SW1	RTC_WAKE_UP2	J23-6	D5 / PWM

DA16200 DA16600 Getting Started with AWS® IoT Core

Function	DA16200 EVK		DA16600 EVK		EK-RA6M4 Board	
	Pin Number	Pin Name	Pin Number	Pin Name	Pin Number	Pin Name
MCU_WAKE_UP	J4.18	GPIOA_11	J2.9	GPIOA_11	None	None

4.2.2.1 UART Connection for AT Commands

The default configuration of UART1 (DA16200 EVB) or UART2 (DA16600 EVB) for AT commands is shown in Table 2.

Table 2: Default Configuration for UART1 or UART2

Settings	Value
Baud Rate	115200
Data Bits	8
Parity	None
Stop Bits	1
Flow Control (HW/SW)	None

The DA16200 EVB uses GPIOA_4 and GPIOA_5 for UART1 TX and UART1 RX, and the DA16600 EVB uses GPIOC_6 and GPIOC_7 for UART2 TX and UART2 RX by default. In addition, GND needs to be connected to the host MCU as shown in Figure 59.

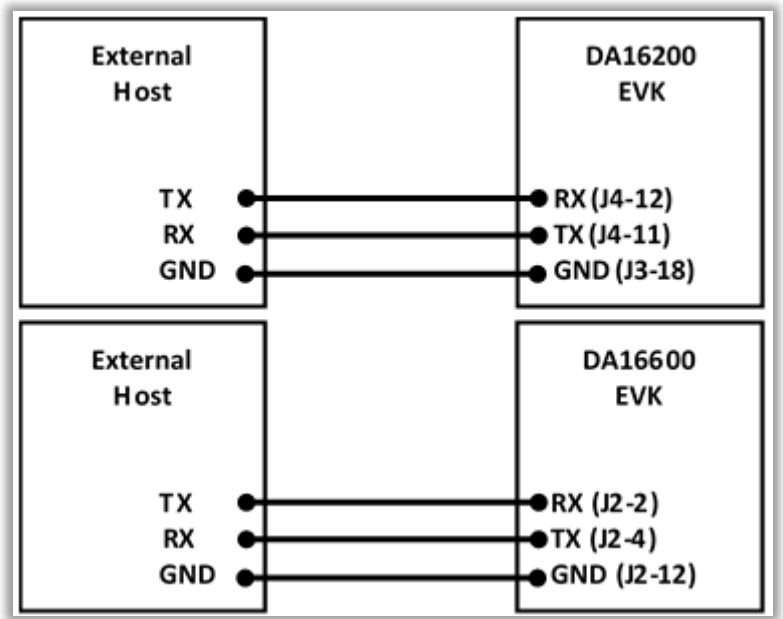


Figure 59: Default UART HW Connection

If GPIO pin configuration is changed using AT commands, other connections for UART1 can be used as shown in Figure 60. The following AT command is used for GPIOA_2 for UART1 TX and GPIOA_3 for UART1 RX. Table 3 shows the pin combination for UART1.

```
AT+AWS=SET NV_PIN_MUX BMUX_UART1d // GPIOA_2 and GPIOA_3 for UART1
```

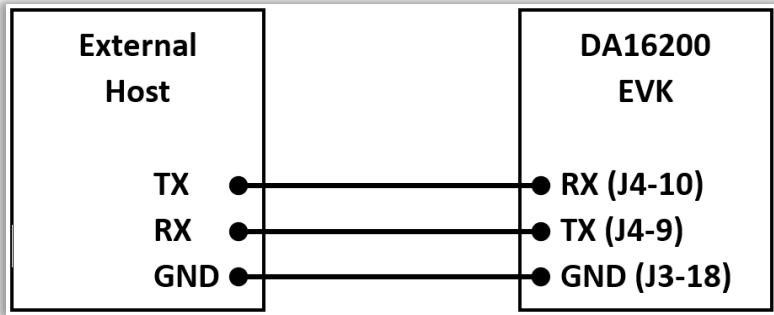


Figure 60: Sample Example of UART1 Connection

Table 3: UART1 Pin Configuration

PIN Mux	GPIO	Signal Name
PIN_AMUX	GPIOA_0	TX
	GPIOA_1	RX
PIN_BMUX	GPIOA_2	TX
	GPIOA_3	RX
PIN_CMUX	GPIOA_4	TX
	GPIOA_5	RX
PIN_DMUX	GPIOA_6	TX
	GPIOA_7	RX

When Dynamic Power Management (DPM) mode is enabled and DA16200/DA16600 is in DPM LPM, the host MCU must wake up the DA16200/DA16600 from DPM LPM using RTC_WAKE_UP. Then, the host MCU can send or receive data over the network in DPM Fully Functional Mode (FFM). The wakeup event is triggered when the GPIO pin of the host MCU changes from Low to High and then back to Low.

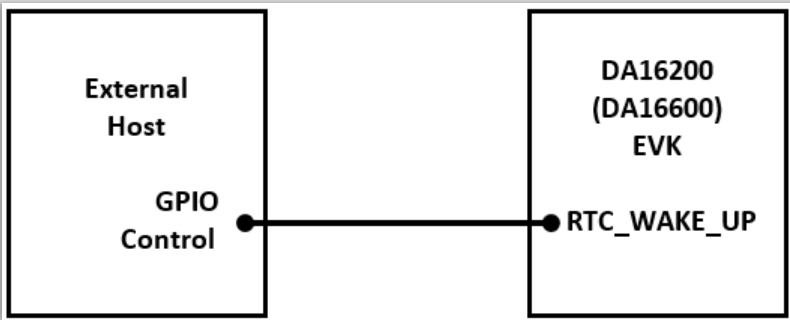


Figure 61: HW Connection for Waking Up DA16200/DA16600

The host MCU may be in sleep mode when DA16200/DA16600 wakes up from DPM LPM and needs to send responses to the host MCU. In this scenario, the DA16200/DA16600 needs to wake up the host MCU from sleep using GPIO as shown in [Figure 61](#). This connection is not required if the host MCU does not use sleep mode. GPIOA_11 is available on DA16200/DA16600 EVB for waking up host MCU by default (see [Figure 62](#)) and it can be configured using the following AT commands.

DA16200 DA16600 Getting Started with AWS® IoT Core

```
AT+AWS SET APP_MCU_WKAEUP_PORT GPIO_UNIT_A // GPIO_A port
AT+AWS SET APP_MCU_WKAEUP_PIN GPIO_PIN11 // GPIO_11
```

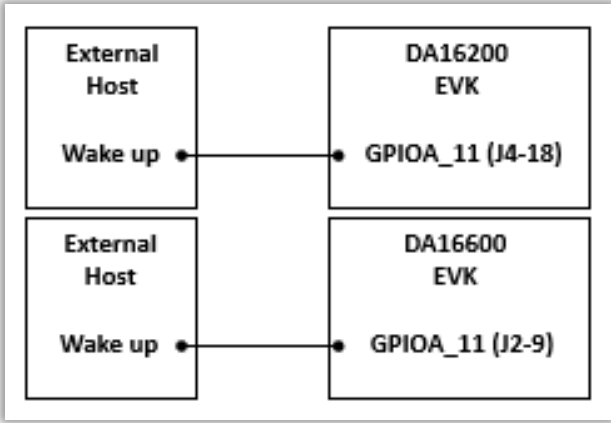


Figure 62: Default Pin Configuration for Waking Up Host MCU

Other GPIOs in the DA16200 EVB can be used for waking up the host MCU as shown in [Table 4](#). For example, GPIOC_6 can be configured for waking up host MCU using the below AT commands (see [Figure 63](#)).

```
AT+AWS SET APP_MCU_WKAEUP_PORT GPIO_UNIT_C // GPIO_C port
AT+AWS SET APP_MCU_WKAEUP_PIN GPIO_PIN6 // GPIO_6
```

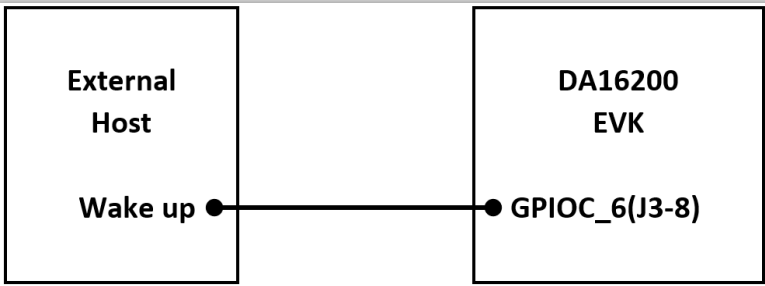


Figure 63: Another Pin Configuration for Waking up Host MCU

Table 4: GPIO Pin Configuration

Port	PIN Mux	GPIO
GPIO_UNIT_A	PIN_AMUX	GPIOA_0
		GPIOA_1
	PIN_BMUX	GPIOA_2
		GPIOA_3
	PIN_CMUX	GPIOA_4
		GPIOA_5
	PIN_DMUX	GPIOA_6
		GPIOA_7

DA16200 DA16600 Getting Started with AWS® IoT Core

Port	PIN Mux	GPIO
	PIN_EMUX	GPIOA_8
		GPIOA_9
	PIN_FMUX	GPIOA_10
		GPIOA_11
GPIO_UNIT_C	PIN_UMUX	GPIOC_6
		GPIOC_7
		GPIOC_8

4.2.3 Programming Firmware Images for DA16200/DA16600

When using an EVB for the first time, the firmware must be updated to the latest version. See Ref [3] for details. After programming the firmware image, factory reset is required to enter the AWS IoT configuration setting mode. This can be done by pushing the “Factory_RST” button for 5 seconds as shown in Figure 64 and Figure 65, and the logs from DA16200/DA16600 are shown in the box below.

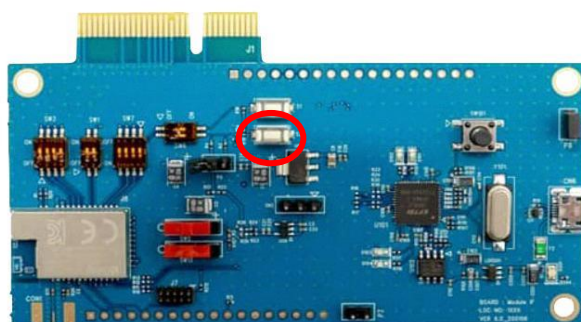


Figure 64: Factory Reset Button on DA16200 EVB

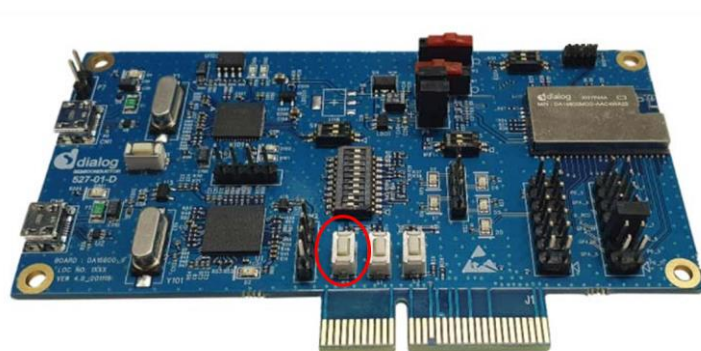


Figure 65: Factory Reset Button on DA16600 EVB

DA16200 DA16600 Getting Started with AWS® IoT Core

```
[DA16200]#
Factory reset ready.

Factory Resetting....

DA16200 concurrent factory reset AP mode = 1 ("AP_ONLY")....

....[app_set_customer_ap_configure] set AP config mode = 0

apps_reboot_ap_mode  Customer configuration ...
.
  default_ssid = "Dialog_DA16200" ..., ap_config_param->ssid_name

  PW = 1234567890

  PW = 1234567890  completed
.
apps_reboot_ap_mode  IPADDR_CUSTOMER...
....
apps_reboot_ap_mode  customer_dhcpd_flag == DHCPD_CUSTOMER..
.....
OK
```

DA16200 DA16600 Getting Started with AWS® IoT Core

```
[/DA16600]#
Factory reset ready.

Factory Resetting....
Set STA Mode ...

Rebooting....

Reset BLE ...

Wakeup source is 0x0
[dpm_init_retnmemory] DPM INIT CONFIGURATION(1)

*****
*          DA16600 SDK Information
* -----
*
* - CPU Type       : Cortex-M4 (120 MHz)
* - OS Type        : FreeRTOS 10.4.3
* - Serial Flash   : 4 MB
* - SDK Version    : V3.2.8.0 AWS-ATCMD Doorlock Ref. QFN GEN
* - F/W Version    : FRTOS-GEN01-01-f017bdfd51-006558
* - F/W Build Time : Sep  5 2023 17:17:05
* - Boot Index     : 0
*
*****

gpio wakeup enable 00000402
[combo] dpm_boot_type = 0

>>> UART1 : Clock=80000000, BaudRate=115200
>>> UART1 : DMA Enabled ...
[combo] BLE_BOOT_MODE_0
[combo] BLE FW VER to transfer ....
>>> v 6.0.14.1114.3 (id=1) at bank_1
[combo] BLE FW transfer done

System Mode : Station Only (0)
>>> Start DA16X Supplicant ...
>>> DA16x Supp Ver2.7 - 2022_03
>>> MAC address (sta0) : d4:3d:39:40:72:16
>>> sta0 interface add OK
>>> Start STA mode...
by default, rf_meas_btcoex(1, 0, 0)

>>> UART2 : Clock=80000000, BaudRate=115200
>>> UART2 : DMA Enabled ...
[UART ready notification]
<<< GAPM_DEVICE_READY_IND
AWS IoT dev_name="DA16200", len=7
IoT dev_name="DA16200", len=7
[combo] Advertising...

[/DA16600] #
```

After the factory reset, the DA16200/DA16600 is now ready to enter the AWS IoT Configuration Settings.

4.2.4 Configure Components for Testing

The following information are required for testing the application with AWS IoT server:

- Unique thingname

The information can be set in the source code for host MCU or using the provided scripts in the downloaded package. See Ref. [3] for how to run the macro script. The scripts are located in the following location.

```
\DA16x00_img\script\doorlock.ttl
```

DA16200 DA16600 Getting Started with AWS® IoT Core

```
;In order to use this script on DA16200, the console should be prompt
;after setting the DA16200 to STA mode, SNTP client enable, and no DPM mode in easy setup through the
console.
;set configurations with DA16200's console

;set features
sendln "user"
;set board type
sendln "SET APP_BOARD_FEATURE EVK"
mpause 400
;set your thingname
;setln "SET APP_THINGNAME FAE-DOORLOCK-4"
mpause 400
;set broker address
sendln "SET AWS_BROKER alkzdt4nun8bnh-ats.iot.ap-northeast-2.amazonaws.com"
mpause 400
```

DA16200 DA16600 Getting Started with AWS® IoT Core

The MCU source code can be found in the following file:

\\MCU\\RA6M4\\Src\\atcmd\\at_cmd.c.

```
#define MAX_RETRY_SEND_COUNT      10

/* AWS features, configurations, and certification keys */
const char* cmd_set_cfg[MAX_CFG_NUM] =
{
    "\\r\\nAT+\"PLATFORM\" SET AWS_USE_FP 0\\r\\n",
    "\\r\\nAT+\"PLATFORM\" SET APP_BOARD_FEATURE EVK\\r\\n",
    "\\r\\nAT+\"PLATFORM\" SET APP_THINGNAME FAE-DOORLOCK-4\\r\\n",
    "\\r\\nAT+\"PLATFORM\" SET AWS_BROKER alkzdt4nun8bnh-ats.iot.ap-northeast-2.amazonaws.com\\r\\n",
    "\\r\\nAT+\"PLATFORM\" SET APP_LPORT 1883\\r\\n",
    "\\r\\nAT+\"PLATFORM\" SET APP_SUBTOPIC /AppControl\\r\\n",
    "\\r\\nAT+\"PLATFORM\" SET APP_PUBTOPIC /DeviceControl\\r\\n",
    "\\r\\nAT+\"PLATFORM\" CFG 0 app_door 1 2\\r\\n",           /* mcu sub.   str */
    "\\r\\nAT+\"PLATFORM\" CFG 1 mcu_door 1 0\\r\\n",       /* mcu pub.   str */
    "\\r\\nAT+\"PLATFORM\" CFG 2 app_window 1 2\\r\\n",     /* mcu sub.   str */
    "\\r\\nAT+\"PLATFORM\" CFG 3 mcu_window 1 0\\r\\n",     /* mcu pub.   str */
    "\\r\\nAT+\"PLATFORM\" CFG 4 battery 0 1\\r\\n",        /* shadow    int */
    "\\r\\nAT+\"PLATFORM\" CFG 5 temperature 2 1\\r\\n",    /* shadow     float */
    "\\r\\nAT+\"PLATFORM\" CFG 6 doorStat 1 1\\r\\n",       /* shadow     str */
    "\\r\\nAT+\"PLATFORM\" CFG 7 windowStat 1 1\\r\\n",     /* shadow     str */
    "\\r\\nAT+\"PLATFORM\" CFG 8 app_shadow 1 2\\r\\n",     /* mcu sub.   str */
    "\\r\\nAT+\"PLATFORM\" CFG 9 mcu_shadow 1 0\\r\\n",     /* mcu pub.   str */
    "\\r\\nAT+\"PLATFORM\" SET SLEEP_MODE 3\\r\\n",
    "\\r\\nAT+\"PLATFORM\" SET USE_DPM 1\\r\\n",
    "\\r\\nAT+\"PLATFORM\" SET RTC_TIME 1740\\r\\n",
    "\\r\\nAT+\"PLATFORM\" SET DPM_KEEP_ALIVE 30000\\r\\n",
    "\\r\\nAT+\"PLATFORM\" SET USE_WAKE_UP 0\\r\\n",
    "\\r\\nAT+\"PLATFORM\" SET TIM_WAKE_UP 10\\r\\n",
    "\\r\\nAT+\"PLATFORM\" SET APP_MCU_WKAEUP_PORT GPIO_UNIT_A\\r\\n", /* GPIO_UNIT_A or GPIO_UNIT_C */
    "\\r\\nAT+\"PLATFORM\" SET APP_MCU_WKAEUP_PIN GPIO_PIN11\\r\\n" /* GPIO_PIN0 ~ GPIO_PIN11 or
GPIO_PIN6~GPIO_PIN8 */
};
```

4.2.5 Test without Host MCU

If the host MCU is not available, the AWS IoT commands can be tested with the script provided in the downloaded package.

Door lock for two-way communication: \\DA16x00_img\\script\\doorlock.ttl.

NOTE

The example script only supports initial value setting. To fully verify the operation of the AT commands, the host MCU should be used for interacting with the server and application.

DA16200 DA16600 Getting Started with AWS® IoT Core

4.2.6 Test with Host MCU

The e²studio is required for building source code for host MCU and programming the images to host MCU. Visit the Renesas website (<https://www.renesas.com/us/en/software-tool/e-studio>) for downloading and installing the e²studio. After installing the e²studio, complete the following steps for building and programming.

1. Import the project file to \MCU\RA6M4\. See Figure 66.

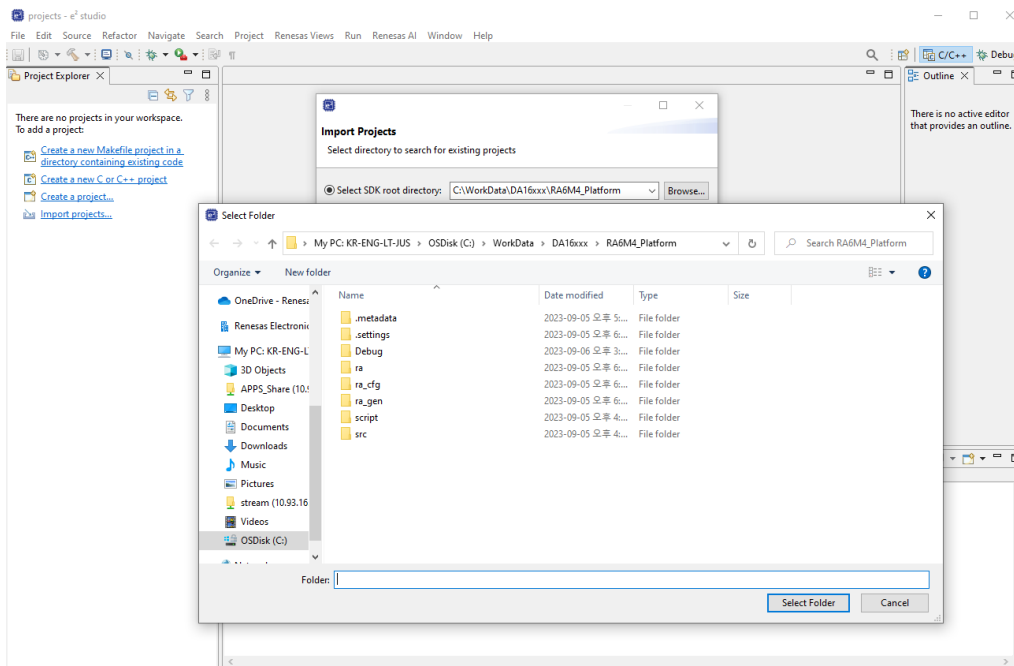


Figure 66: e²studio Project File

NOTE

When connecting to the RA6M4 MCU for the first time or changing the configuration, complete the step 2 to set up the FSP configuration.

2. Select Configurations.xml to set FSP configuration of the RA6M4 MCU (see Figure 67).

DA16200 DA16600 Getting Started with AWS® IoT Core

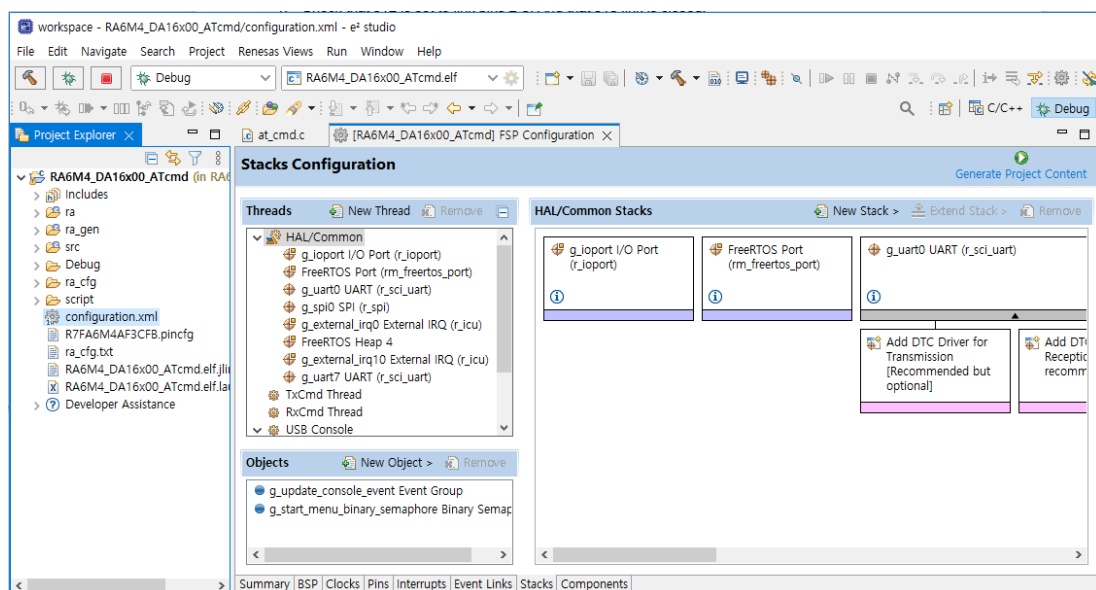


Figure 67: FSP Configuration

3. Use the Thing name received from the FAE to test without setting up a server.
4. Change the Thing name to the received name (see [Figure 68](#)).

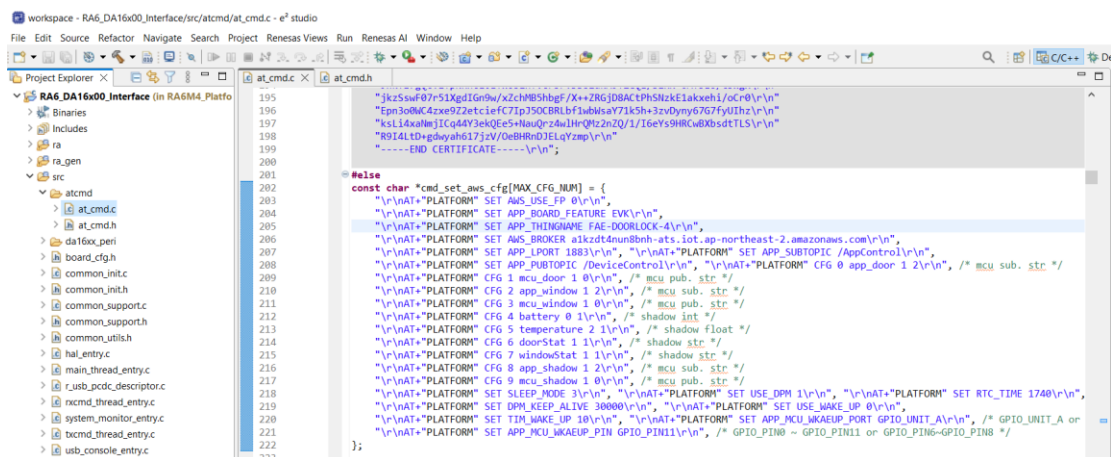


Figure 68: Thing Name in MCU Source Code

5. Select project > Build Project to build (see [Figure 69](#)).

DA16200 DA16600 Getting Started with AWS® IoT Core

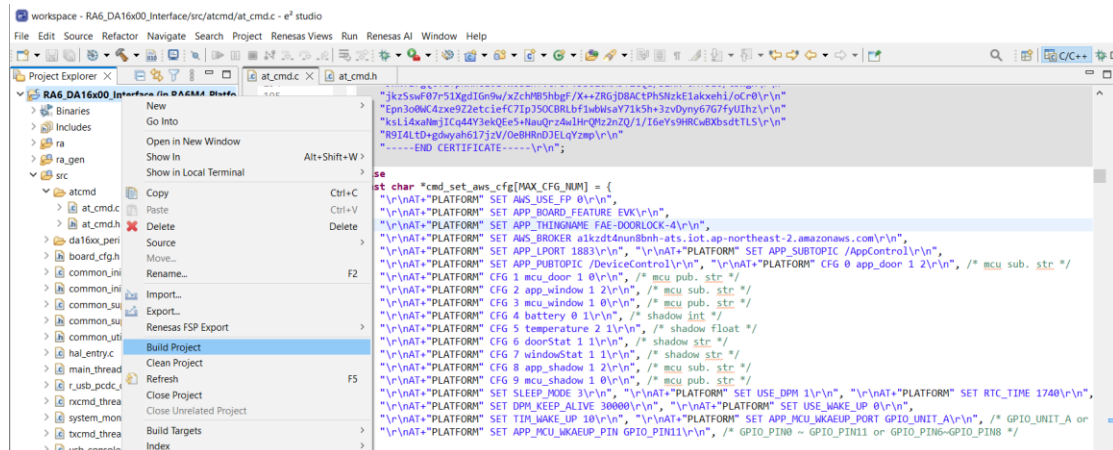


Figure 69: Build Project

6. Select Debug Configurations to set the connection to the RA6M4 MCU (see [Figure 70](#)).

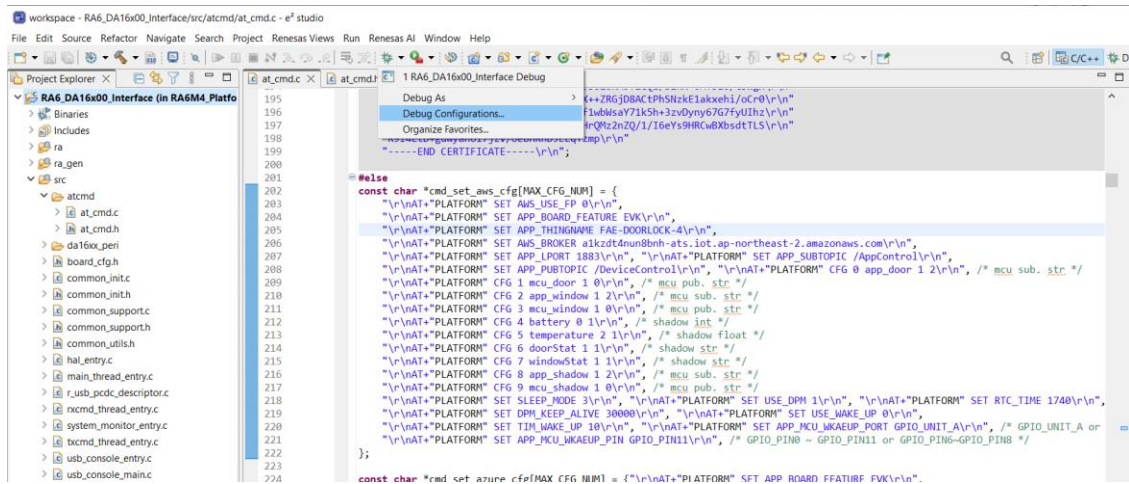


Figure 70: Debug Configurations

7. Selecting the debug tab will open the window shown in [Figure 71](#), and change the configuration as shown and then select the **Apply** and **Debug** buttons.

DA16200 DA16600 Getting Started with AWS® IoT Core

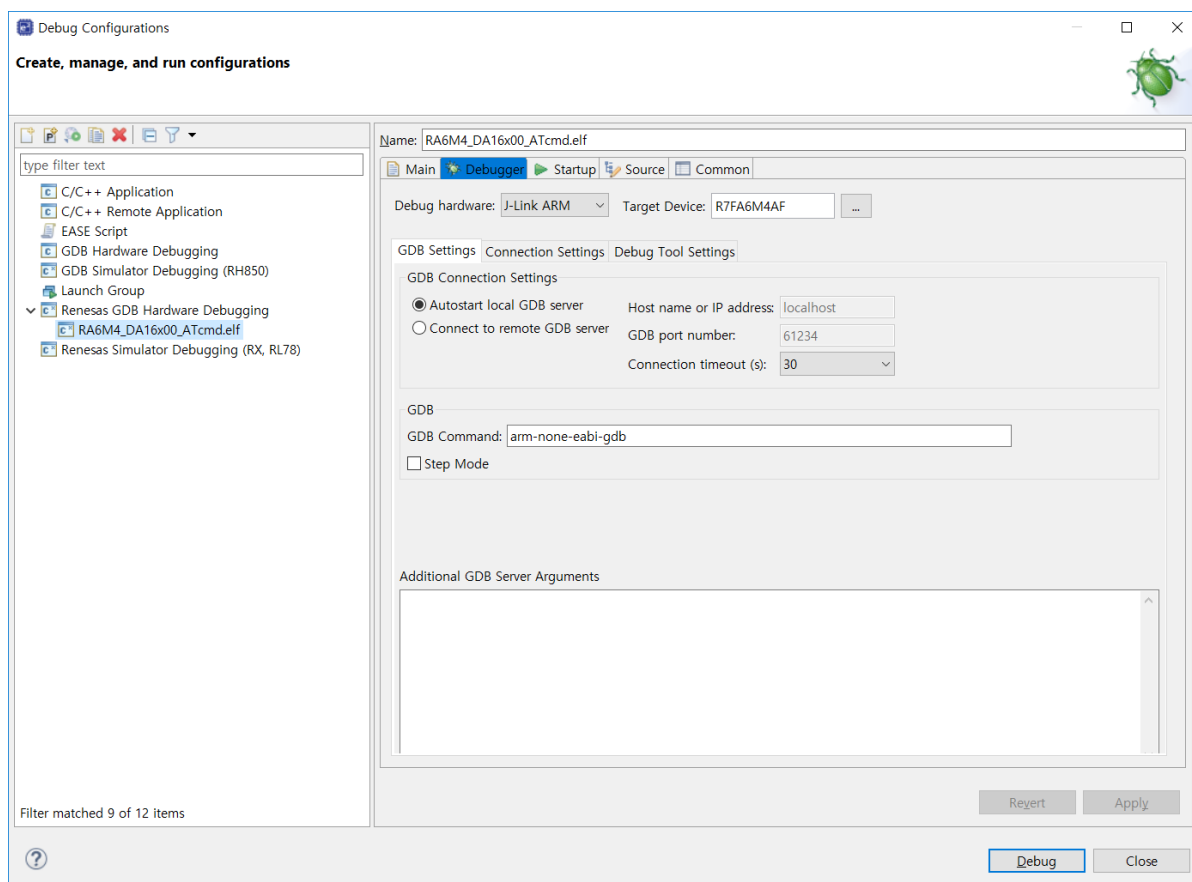


Figure 71: Set Debug Configurations

8. The following shows the console output of the DA16200 after a factory mode reset.

```
Soft-AP is Ready (d4:3d:39:10:d5:07)

>>> UART1 : Clock=80000000, BaudRate=115200
>>> UART1 : DMA Enabled ...
[UART ready notification]
[http_server_task] HTTP-Server Start!!
=====
[ AWS-IOT AT COMMAND ]
[ aws_shadow_dpm auto start]
AWS_IOT on Station Mode for "FAE-DOORLOCK-4"
=====
[pal_app_dpm_auto_start] mcu_wakeup_port=-1, mcu_wakeup_pin=0x0
default set to mcu_wakeup_port=0, mcu_wakeup_pin=0x800

Root CA: X
Certificate: X
Private Key: X

nvram read string(thingname) error
invalid APP feature...can't start APP Platform thread...check again
.. UART ready
```


DA16200 DA16600 Getting Started with AWS® IoT Core

9. The following shows the console output of the DA16200 when setting the AWS IoT configuration via AT commands from an MCU.

```
=====
argc num = 2
argv[0]: AT+AWS
argv[1]: CFG 3 mcu_window 1 0
=====
```

```
=====
Att[3] number   : 3
Att[3] name     : mcu_window
Att[3] data type: 1
Att[3] MQTT type: 0
=====
```

```
=====
argc num = 2
argv[0]: AT+AWS
argv[1]: CFG 4 battery 0 1
=====
```

```
=====
Att[4] number   : 4
Att[4] name     : battery
Att[4] data type: 0
Att[4] MQTT type: 1
=====
```

```
=====
argc num = 2
argv[0]: AT+AWS
argv[1]: CFG 5 temperature 2 1
=====
```

```
=====
Att[5] number   : 5
Att[5] name     : temperature
Att[5] data type: 2
Att[5] MQTT type: 1
=====
```

DA16200 DA16600 Getting Started with AWS® IoT Core

10. The following shows the console output of the DA16200 after the Soft AP has been configured and it is waiting to be provisioned by the mobile application.

```
Soft-AP is Ready (d4:3d:39:10:d5:07)

>>> UART1 : Clock=80000000, BaudRate=115200
>>> UART1 : DMA Enabled ...
[UART ready notification]
[http_server_task] HTTP-Server Start!!
=====
[ AWS-IOT AT COMMAND ]
[ aws_shadow_dpm_auto_start]
AWS_IOT on Station Mode for "FAE-DOORLOCK-4"
=====
[pal_app_dpm_auto_start] mcu_wakeup_port=0, mcu_wakeup_pin=0x800

Root CA: 0
Certificate: 0
Private Key: 0

subscribe index=0, name=app_door
subscribe index=2, name=app_window
newNode index=4
newNode index=5
newNode index=6
newNode index=7
subscribe index=8, name=app_shadow
shadow item count = 4, (integer#=1, string#=2, float#=1)
current shadowConut = 4
pkey=windowStat, pdata=test
current shadowConut = 3
pkey=doorStat, pdata=test
current shadowConut = 2
pkey=temperature, pdata=16.500000
current shadowConut = 1
pkey=battery, pdata=2700

AWS_IOT AP Mode  FAE-DOORLOCK-4

+ATPROV=STATUS 1
=====
[Start Provisioning with TCP/TLS] .. Soft AP Mode
=====
[app_provision_switch_client_thread] Create...(status=0) [10]
[app_provision_TCP_server_thread] Create ...
[app_provision_TLS_server_thread] Create TLS...

>>> Start Provisioning Server (TLS) ...
Wait Accept (TLS)...
[app_find_home_ap] Wi-Fi Scan request success.
[app_find_home_ap:518] (0) iptime_justin / 3 / -34 / 2447
[app_find_home_ap:518] (1) AP-101-201 / 3 / -66 / 2432
[app_find_home_ap:518] (2) SK Wi-FiGIGA551A 2.4G / 3 / -78 / 2422
[app_find_home_ap:518] (3) SK Wi-FiGIGA551A / 3 / -79 / 2422
[app_find_home_ap:518] (4) SK Wi-Fi3801 / 3 / -94 / 2412
[app_find_home_ap:518] (5) NIS-HomeAP11N / 0 / -74 / 2447
[app_provision_TCP_server_thread] socket().. status=1
Wait Accept...
```

4.3 Mobile App Demo

Install the mobile application by searching for **DA16200** or **DA16600** on the Google Play Store or Apple App Store on the mobile devices.

DA16200 DA16600 Getting Started with AWS® IoT Core

4.3.1 Open Door

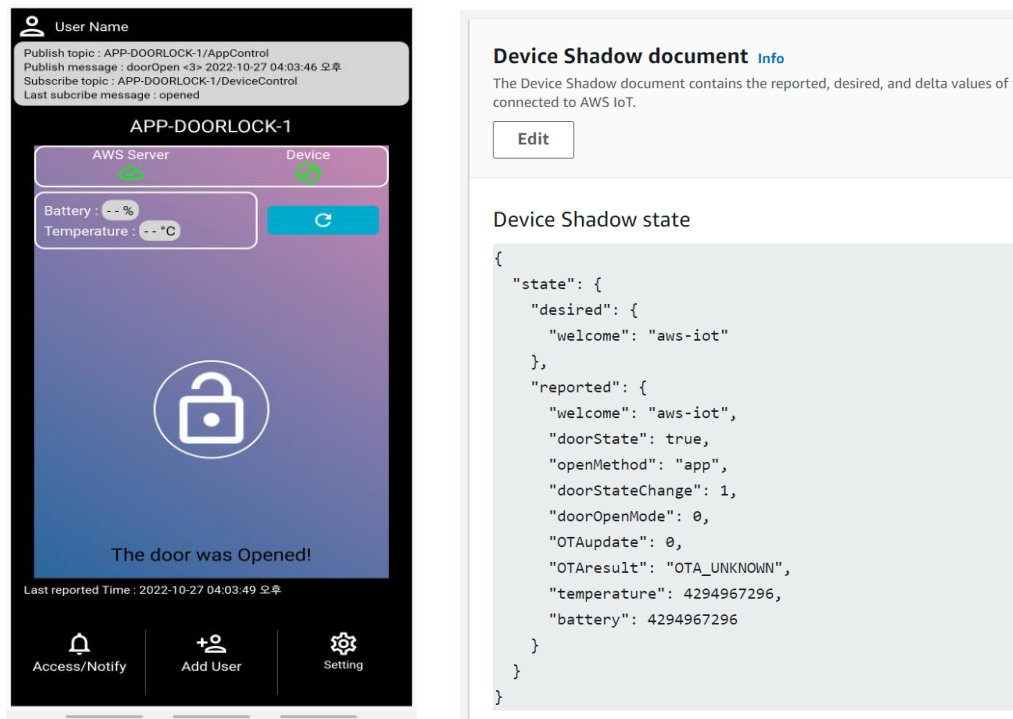


Figure 72: Opened Status on Application and AWS IoT Console

- [Current Status]
 - **Opened**, Battery: __%, Temperature: __ °C (Real values are displayed on door lock ref. board)
 - Mobile APP (User): **Opened image button**
 - AWS (Server)
 - “doorState”: **true**
 - “temperature”: 4294967296
 - “battery”: 4294967296

NOTE
A value of 4294967296 for the temperature or battery fields indicates the function is not available.

- DA16200 (Thing): The status of the device is displayed as shown in the **red text**.

```
[INFO] [DoorLockDemo] [prvEventCallback:728]
Incoming Publish Topic Name: (Command) APP-DOORLOCK-1/AppControl matches subscribed topic.
Incoming Publish Message : doorOpen

open comm
[openControl]

[INFO] [DoorLockDemo] [controlDoorLock:1555] publish (command response) OK - payload: "opened"
DEBUG: [aws_dpm_app_door_work:1974] previous MQTT result = 0, doorLock CMD (=1: 0-idle, 1-open, 2-close, 3-
auto close)

=====

[INFO] [DoorLockDemo] [aws_dpm_app_door_work:2030] publish (shadow doorlock update) OK - payload:
{"state":{"reported":{"doorState":true,"openMethod":"app","doorStateChange":1,"doorOpenMode":0,"OTAupdate":0,
"OTAresult":"OTA_UNKNOWN"}}}
```

DA16200 DA16600 Getting Started with AWS® IoT Core

```
*****
last user Timer ID = 5
last doorOpenFlag state: "true"
last FOTA Stat: 0
last FOTA Url: ""
```

4.3.2 Close Door

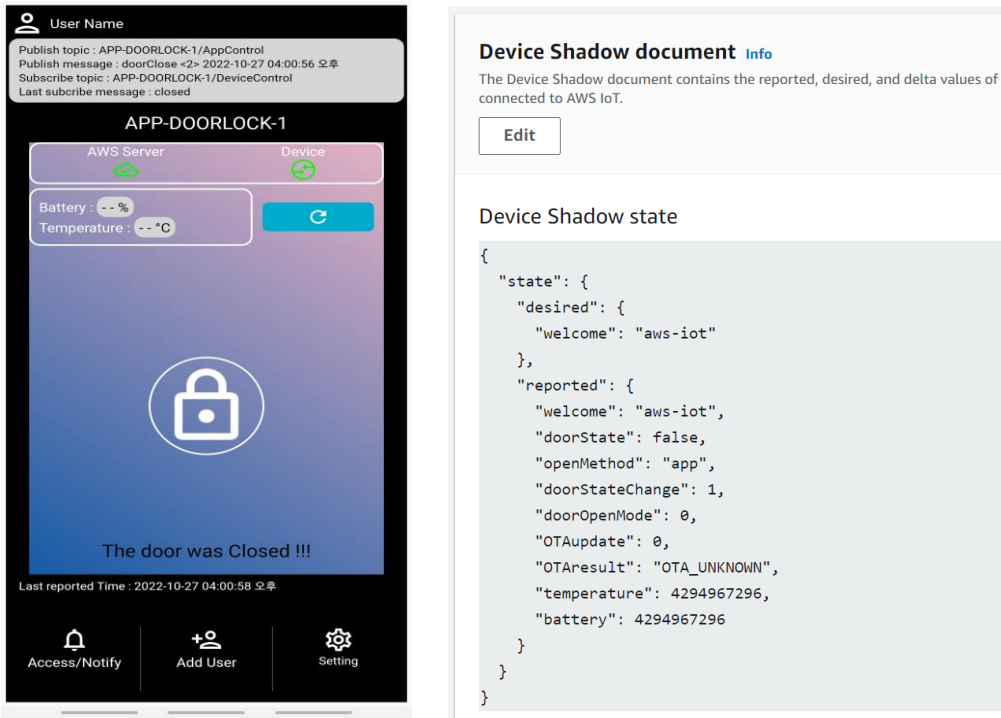


Figure 73: Closed Status on Application and AWS IoT Console

- [Current Status]
 - **Closed**, Battery: __%, Temperature: __ °C (Real values are displayed on door lock ref. board)
 - Mobile APP (User): **Closed image button**
 - AWS (Server)
 - “doorState”: **false**
 - “temperature”: 4294967296
 - “battery”: 4294967296

NOTE
A value of 4294967296 for the temperature or battery fields indicates the function is not available.

- DA16200 (Thing): The status of the device is displayed as shown in the **red text**.
- ```
[INFO] [DoorLockDemo] [prvEventCallback:728]
Incoming Publish Topic Name: (Command) APP-DOORLOCK-1/AppControl matches subscribed topic.
Incoming Publish Message : doorClose

close comm
[closeControl]

[INFO] [DoorLockDemo] [controlDoorLock:1555] publish (command response) OK - payload: "closed"
DEBUG: [aws_dpm_app_door_work:1974] previous MQTT result = 0, doorLock CMD (=2: 0-idle, 1-open, 2-close, 3-
auto close)
```

---

**DA16200 DA16600 Getting Started with AWS® IoT Core**

---

```
[INFO] [DoorLockDemo] [aws_dpm_app_door_work:2030] publish (shadow doorlock update) OK - payload:
{"state":{"reported":{"doorState":false,"openMethod":"app","doorStateChange":1,"doorOpenMode":0,"OTAupdate":0
,"OTAresult":"OTA_UNKNOWN"}}}

last user Timer ID = 5
last doorOpenFlag state: "false"
last FOTA Stat: 0
```

## DA16200 DA16600 Getting Started with AWS® IoT Core

## 5 OTA Update

Over the Air (OTA) is the process of updating the DA16200/DA16600 firmware image through Wi-Fi using an AWS S3 bucket.

To set up OTA update, see [Figure 74](#).

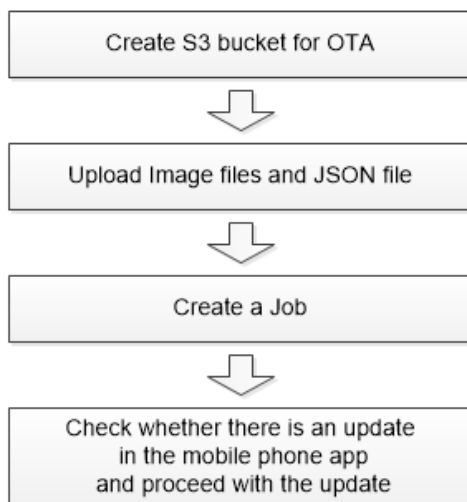


Figure 74: OTA Update

### 5.1 Create S3 Bucket

For OTA update, a new bucket should be created in S3.

1. In the Amazon S3 console, click **Create bucket**. See [Figure 75](#).

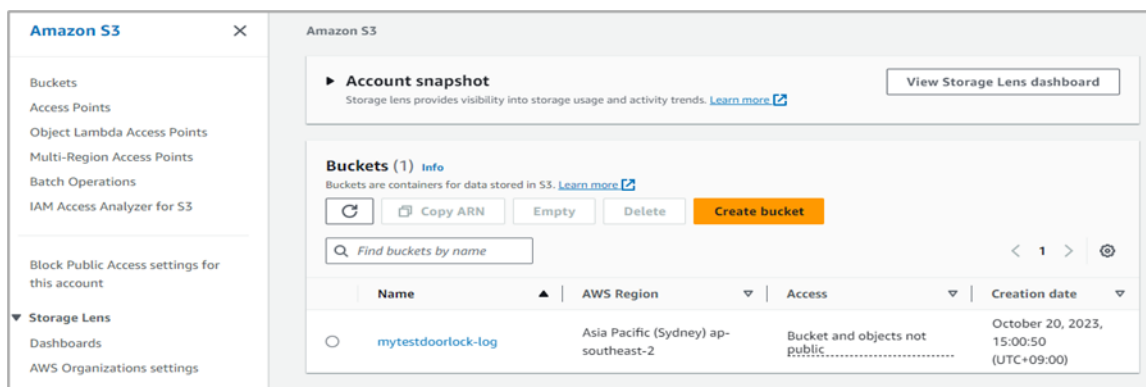


Figure 75: Create Bucket for OTA Update

2. Enter a Bucket name and click **Create bucket**. See [Figure 76](#).

DA16200 DA16600 Getting Started with AWS® IoT Core

Amazon S3 > Buckets > Create bucket

Create bucket 

Info

Buckets are containers for data stored in S3. [Learn more](#)

General configuration

Bucket name

mytest-ota

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

AWS Region

Asia Pacific (Seoul) ap-northeast-2

Copy settings from existing bucket - optional

Only the bucket settings in the following configuration are copied.

Choose bucket

Object Ownership 

Info

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

☐ ACLs disabled (recommended)

All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

☒ ACLs enabled

Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

We recommend disabling ACLs, unless you need to control access for each object individually or to have the object writer own the data they upload. Using a bucket policy instead of ACLs to share data with users outside of your account simplifies permissions management and auditing.

Object Ownership

☐ Bucket owner preferred

If new objects written to this bucket specify the bucket-owner-full-control canned ACL, they are owned by the bucket owner. Otherwise, they are owned by the object writer.

☒ Object writer

The object writer remains the object owner.

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. 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 this bucket 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 or access point policies

S3 will block new bucket and access point 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 or access point policies

S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Bucket Versioning

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. [Learn more](#)

Bucket Versioning

☒ Disable

☐ Enable

User Manual

Revision 1.5

Jan. 26, 2024

CFR0012

67 of 94

© 2024 Renesas Electronics

DA16200 DA16600 Getting Started with AWS® IoT Core

Tags - optional (0)  
You can use bucket tags to track storage costs and organize buckets. [Learn more](#)

No tags associated with this bucket.

Add tag

Default encryption Info  
Server-side encryption is automatically applied to new objects stored in this bucket.

Encryption type Info

☒ Server-side encryption with Amazon S3 managed keys (SSE-S3)

☐ Server-side encryption with AWS Key Management Service keys (SSE-KMS)

☐ Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)  
Secure your objects with two separate layers of encryption. For details on pricing, see [DSSE-KMS pricing](#) on the [Storage](#) tab of the [Amazon S3 pricing page](#).

Bucket Key  
Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. [Learn more](#)

☒ Disable

☐ Enable

Advanced settings

After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.

CancelCreate bucket

Figure 76: Bucket Name for OTA Update

3. Select the created bucket from the list. See Figure 77.

Successfully created bucket "mytest-ota"  
To upload files and folders, or to configure additional bucket settings, choose [View details](#).

Amazon S3 > Buckets

Account snapshot  
Storage lens provides visibility into storage usage and activity trends. [Learn more](#)

View Storage Lens dashboard

Buckets (2) Info  
Buckets are containers for data stored in S3. [Learn more](#)

Copy ARN

Empty

Delete

Create bucket

Find buckets by name

< 1 >

|                       | Name               | AWS Region                           | Access                        | Creation date                          |
|-----------------------|--------------------|--------------------------------------|-------------------------------|----------------------------------------|
| <input type="radio"/> | mytest-ota         | Asia Pacific (Seoul) ap-northeast-2  | Bucket and objects not public | October 23, 2023, 16:18:57 (UTC+09:00) |
| <input type="radio"/> | mytestdoorlock-log | Asia Pacific (Sydney) ap-southeast-2 | Bucket and objects not public | October 20, 2023, 15:00:50 (UTC+09:00) |

Figure 77: Created Buckets for OTA

4. Click the **Permissions** tab, and then click **Edit**. See Figure 78.

This bucket must be modified for public access in the next step.

|                                                            |
|------------------------------------------------------------|
| NOTE                                                       |
| Using public buckets is for development environments only. |



DA16200 DA16600 Getting Started with AWS® IoT Core

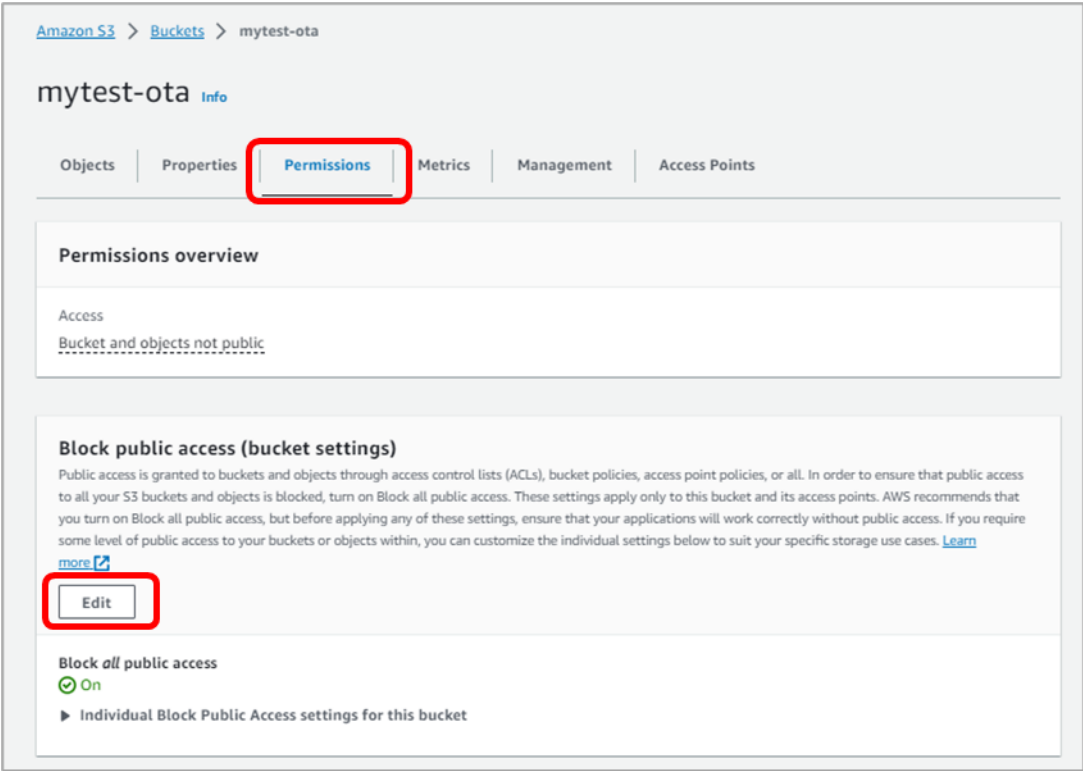


Figure 78: Edit Bucket for Public Access

5. Clear all checkboxes, and then click **Save changes**. See Figure 79.

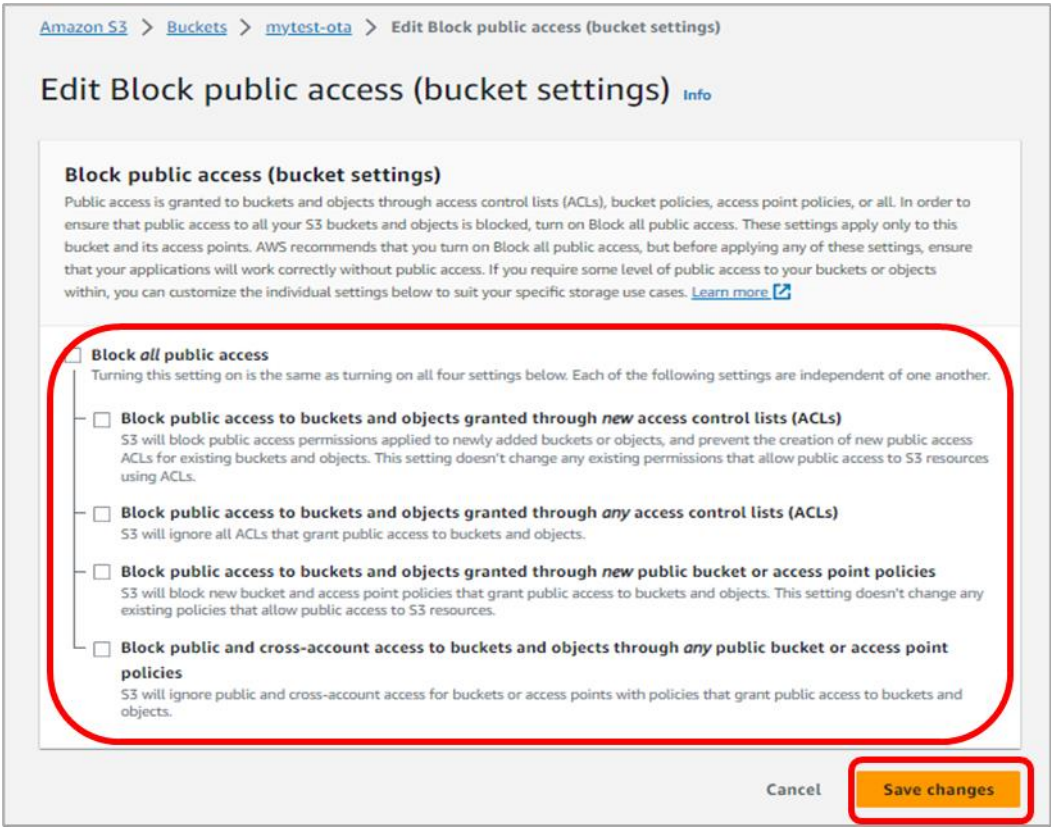


Figure 79: Public Access Settings for Bucket

DA16200 DA16600 Getting Started with AWS® IoT Core

6. To save the settings, enter the word “confirm”, and then click **Confirm**. See Figure 80.

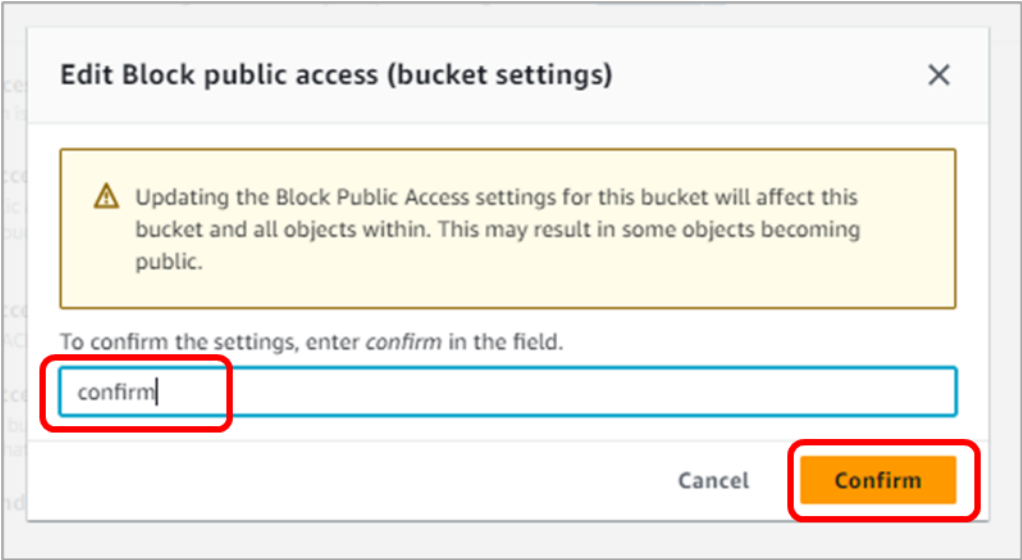


Figure 80: Confirm Settings

7. Check whether all block options of public access are off. See Figure 81.

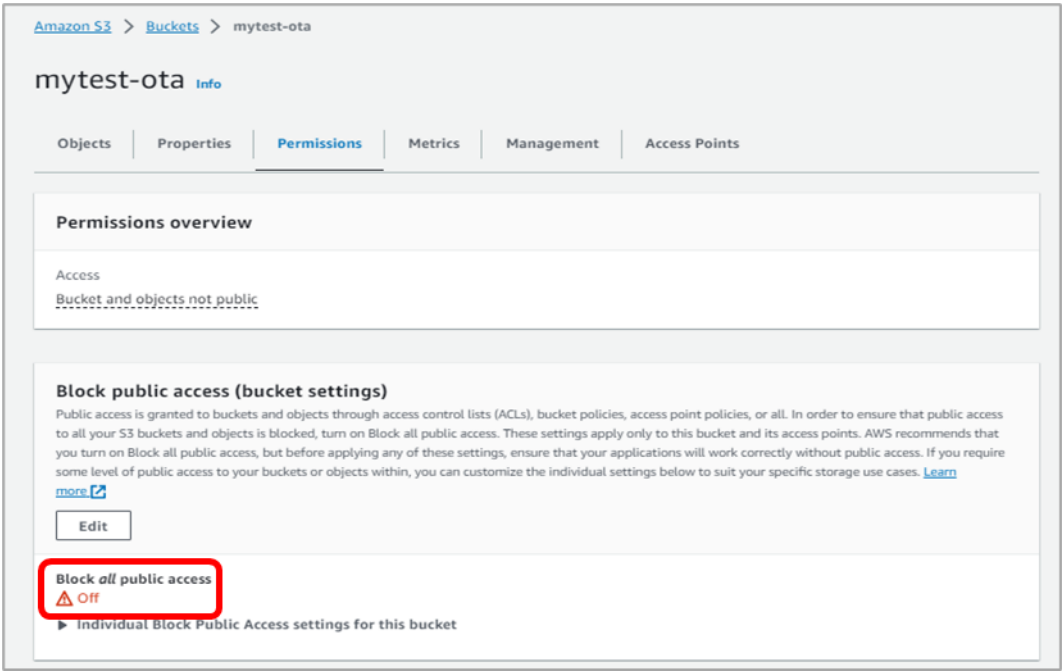


Figure 81: Settings Updated

8. Click **Access Control List (ACL) > Edit** and next to **Everyone**, select **Read** bucket permissions, and then click **Save changes**. See Figure 82.

NOTE

Check the document below on how to avoid ACLs.  
<https://docs.aws.amazon.com/AmazonS3/latest/userguide/about-object-ownership.html>

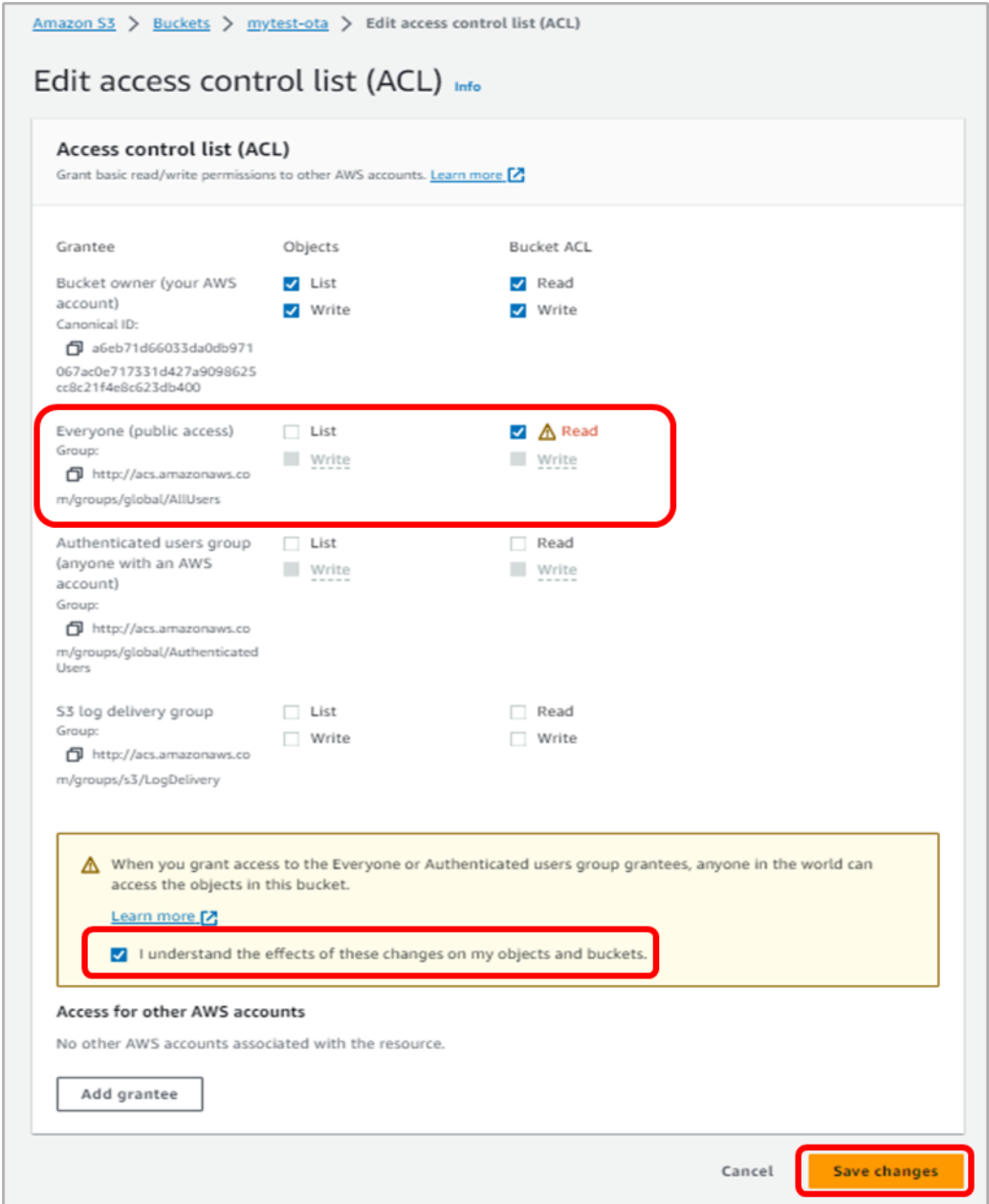


Figure 82: Public Access for Everyone

9. The bucket policy must be added as shown in [Figure 83](#) and [Figure 84](#).  
"User Bucket Name" in [Figure 84](#) is the name of the S3 bucket created for an OTA update.

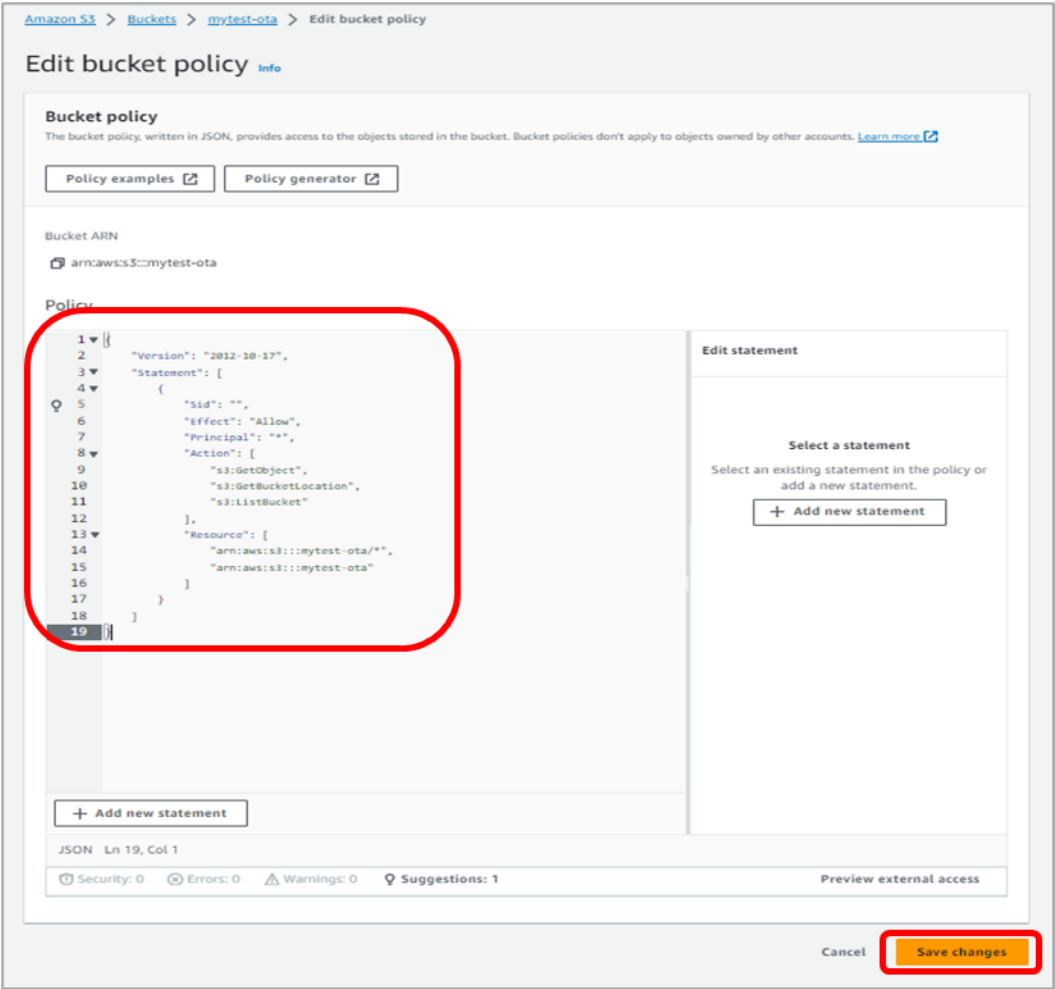


Figure 83: Bucket Policy Editor

```

{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "",
 "Effect": "Allow",
 "Principal": "*",
 "Action": [
 "s3:GetObject",
 "s3:GetBucketLocation",
 "s3:ListBucket"
],
 "Resource": [
 "arn:aws:s3::User Bucket Name/*",
 "arn:aws:s3::User Bucket Name"
]
 }
]
}
```

Figure 84: Bucket Policy JSON

## DA16200 DA16600 Getting Started with AWS® IoT Core

### 5.2 Upload Image File and JSON File

1. Rename the Image files as follows:
  - RTOS Image: DA16200\_FRTOS-GEN01.img
2. Click **Upload** to be able to upload the Image files and JSON file for an OTA update. See [Figure 85](#).

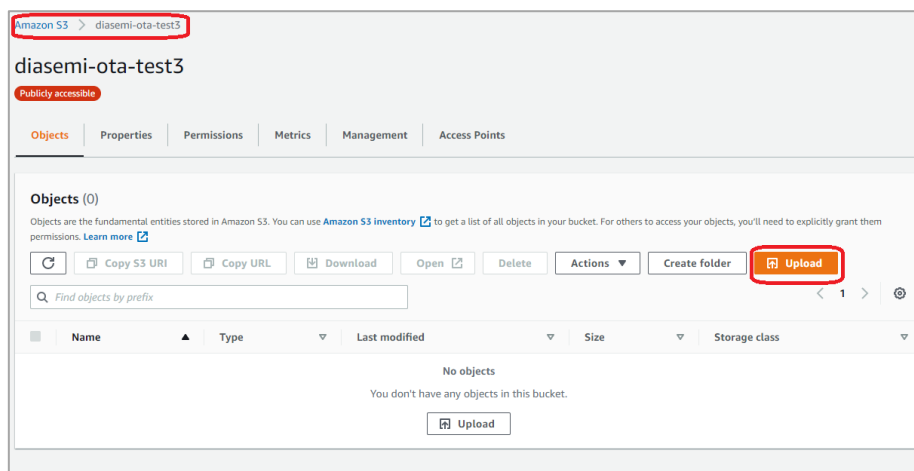


Figure 85: Upload Files

3. Drag and drop or add files to upload.
4. There is one IMG file for a DA16200 OTA update, and the JSON file is a path setting file for the update. The important thing is that the names of the two files for the update should be the same as in [Figure 86](#).

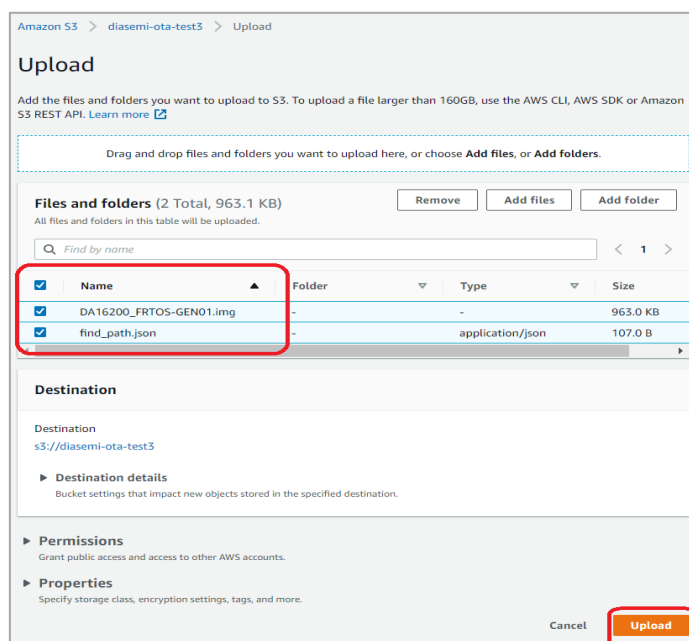


Figure 86: Ready to Upload

The JSON information for an OTA update is as follows:

```
{ "operation": "install",
 "Source": "https:// User Bucket Name.s3.ap-northeast-2.amazonaws.com/" }
```

DA16200 DA16600 Getting Started with AWS® IoT Core

```
{ "operation": "install",
 "Source": "https:// User Bucket Name.s3.ap-northeast-2.amazonaws.com/" }
```

"User Bucket Name" is the name of the S3 bucket created for an OTA update.

The URL policy of the "Source" can be changed by AWS.

5. Click the uploaded file name to check it. See [Figure 87](#).

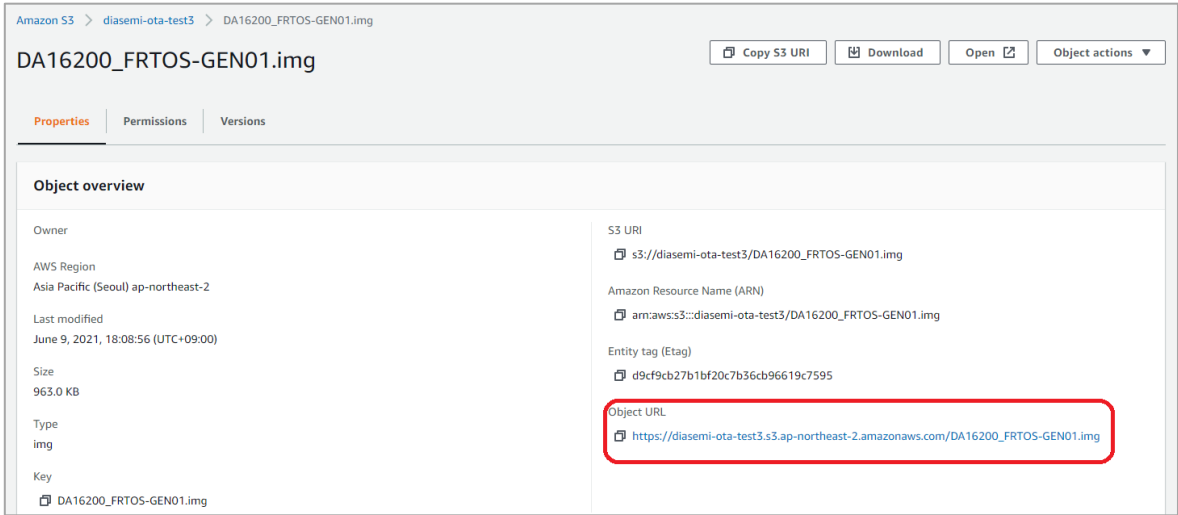


Figure 87: URL of Source

6. Check if the files are uploaded correctly. See [Figure 88](#). The user can delete and/or re-upload files to the bucket in the **Actions** tab.

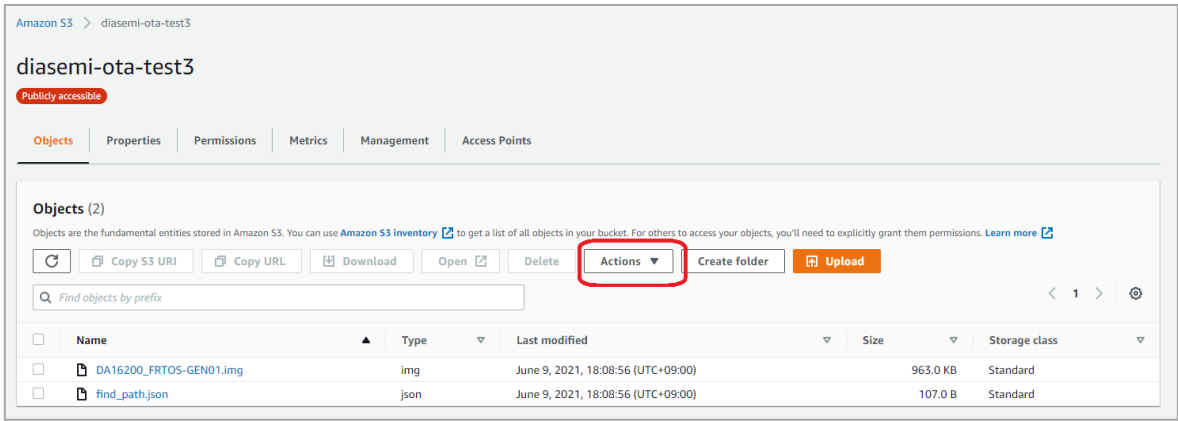


Figure 88: Uploaded Files

The result is that a publicly accessible bucket is created. See [Figure 89](#).



Figure 89: Completed Setup for OTA Update

5.3 Create Job

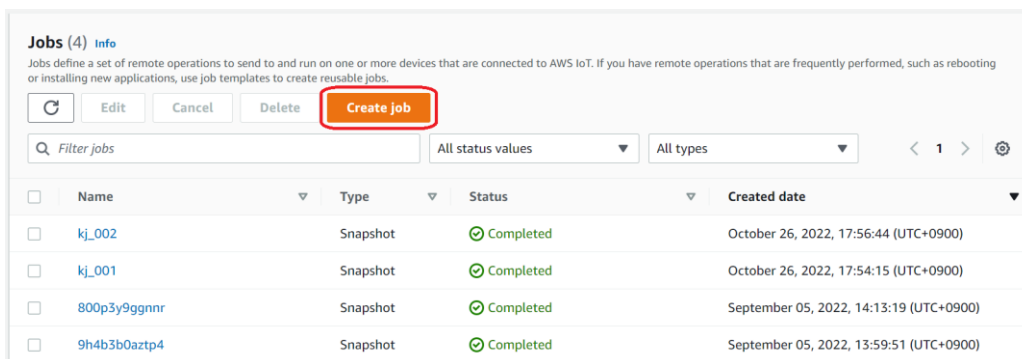
AWS IoT Jobs is a service that allows users to define a set of remote operations that are sent to and executed on one or more devices connected to AWS IoT.

For an OTA update, go to the **IoT Core** service page in AWS Management Console. OTA is the process of replacing a product with a newer version of the same product. A Job must be created

## DA16200 DA16600 Getting Started with AWS® IoT Core

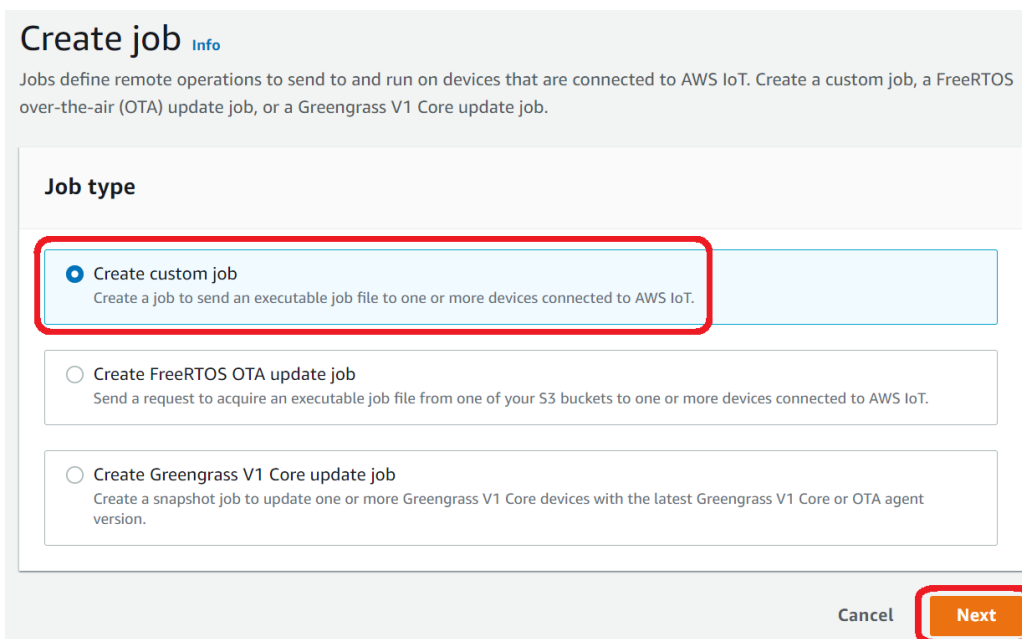
and registered to do an OTA update. It is a task to access the file uploaded to the bucket of the S3 service. If the server operator registers this Job at the desired time, the test Thing will proceed with the OTA update.

1. In the AWS Management Console, go to **IoT core > Manage > Remote Actions > Jobs**.
2. Click **Create job**. See [Figure 90](#). [Figure 90: Create Job](#)



**Figure 90: Create Job**

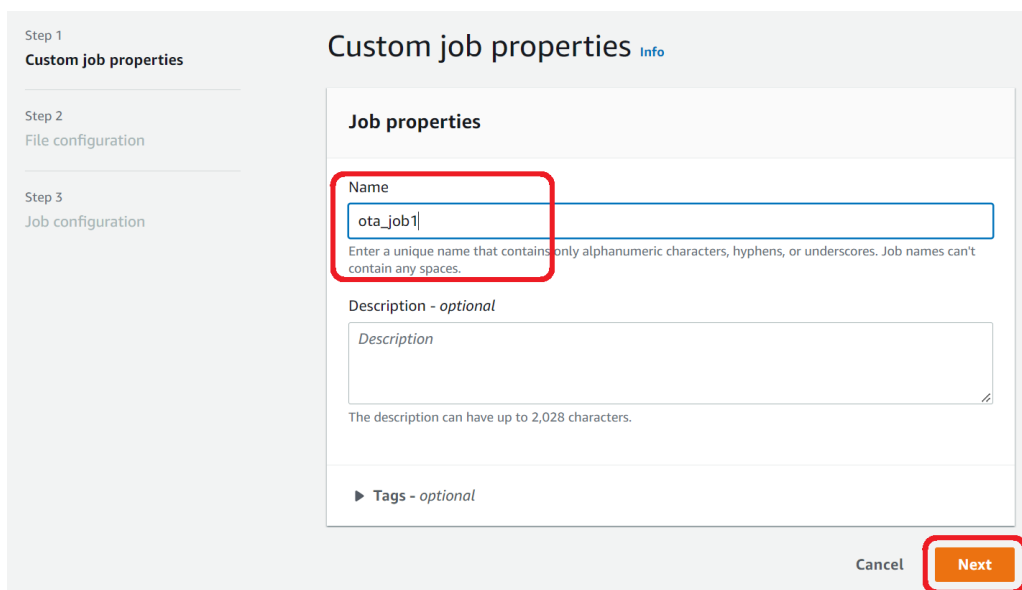
3. Select **Create custom job** and click **Next**. See [Figure 91](#).



**Figure 91: Create Custom Job**

4. Enter the Job name, click **Next**, and then select the devices to update. The Thing to select is available in the list of options. See [Figure 92](#) and [Figure 93](#).

## DA16200 DA16600 Getting Started with AWS® IoT Core



Step 1  
Custom job properties

Step 2  
File configuration

Step 3  
Job configuration

### Custom job properties [Info](#)

#### Job properties

**Name**

ota\_job1

Enter a unique name that contains only alphanumeric characters, hyphens, or underscores. Job names can't contain any spaces.

**Description - optional**

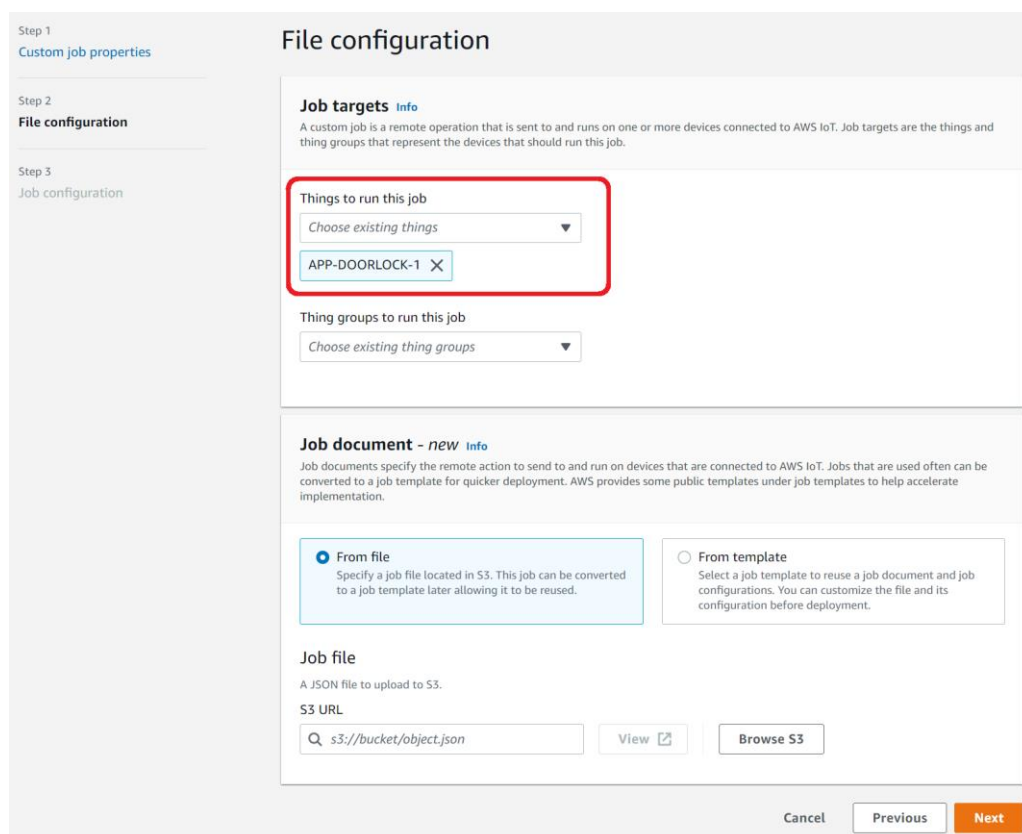
Description

The description can have up to 2,028 characters.

► Tags - optional

Cancel **Next**

Figure 92: Make Job Name



Step 1  
Custom job properties

Step 2  
File configuration

Step 3  
Job configuration

### File configuration

#### Job targets [Info](#)

A custom job is a remote operation that is sent to and runs on one or more devices connected to AWS IoT. Job targets are the things and thing groups that represent the devices that should run this job.

**Things to run this job**

Choose existing things ▼

APP-DOORLOCK-1 ✕

**Thing groups to run this job**

Choose existing thing groups ▼

#### Job document - new [Info](#)

Job documents specify the remote action to send to and run on devices that are connected to AWS IoT. Jobs that are used often can be converted to a job template for quicker deployment. AWS provides some public templates under job templates to help accelerate implementation.


☒ **From file**  
Specify a job file located in S3. This job can be converted to a job template later allowing it to be reused.

☐ **From template**  
Select a job template to reuse a job document and job configurations. You can customize the file and its configuration before deployment.

**Job file**

A JSON file to upload to S3.

**S3 URL**

Q s3://bucket/object.json View 

Browse S3

Cancel Previous **Next**

Figure 93: Select Thing for OTA Update

5. Select the thing to perform. The selected thing is defined for the task to be done. See [Figure 94](#), [Figure 95](#), and [Figure 96](#).



## DA16200 DA16600 Getting Started with AWS® IoT Core

Step 1  
Custom job properties

Step 2  
**File configuration**

Step 3  
Job configuration

### File configuration

**Job targets** Info  
A custom job is a remote operation that is sent to and runs on one or more devices connected to AWS IoT. Job targets are the things and thing groups that represent the devices that should run this job.

Things to run this job  
Choose existing things ▼  
APP-DOORLOCK-1 X

Thing groups to run this job  
Choose existing thing groups ▼

**Job document - new** Info  
Job documents specify the remote action to send to and run on devices that are connected to AWS IoT. Jobs that are used often can be converted to a job template for quicker deployment. AWS provides some public templates under job templates to help accelerate implementation.

☒ **From file**  
Specify a job file located in S3. This job can be converted to a job template later allowing it to be reused.

☐ **From template**  
Select a job template to reuse a job document and job configurations. You can customize the file and its configuration before deployment.

**Job file**  
A JSON file to upload to S3.

S3 URL  
Q s3://diasemi-ota-test3/find\_path.json X View Browse S3

**No pre-signing URL found**  
To use pre-signing URL, a placeholder snippet is required in the job file.

Cancel Previous **Next**

Figure 94: Select JSON for OTA Update

Step 1  
Custom job properties

Step 2  
File configuration

Step 3  
**Job configuration**

### Job configuration

**Job run type**  
Configure how your job will deploy to the job targets.

☒ **Snapshot**  
Your job deploys and completes execution for current devices added to thing groups in current job target list.

☐ **Continuous**  
Your job continues to deploy to devices that are added to the thing groups in the job target list.

**Additional configurations - optional**

- ▶ Rollout configuration
- ▶ Job executions timeout configuration
- ▶ Job executions retry configuration - new
- ▶ Abort configuration

Cancel Previous **Submit**

Figure 95: Job Run Type

## DA16200 DA16600 Getting Started with AWS® IoT Core

**Jobs (5)** [Info](#)

Jobs define a set of remote operations to send to and run on one or more devices that are connected to AWS IoT. If you have remote operations that are frequently performed, such as rebooting or installing new applications, use job templates to create reusable jobs.

[Refresh](#)
[Edit](#)
[Cancel](#)
[Delete](#)
[Create job](#)

| <input type="checkbox"/> | Name         | Type     | Status                            | Created date                            |
|--------------------------|--------------|----------|-----------------------------------|-----------------------------------------|
| <input type="checkbox"/> | ota_job1     | Snapshot | In progress - Rollout in progress | October 27, 2022, 13:19:29 (UTC+0900)   |
| <input type="checkbox"/> | kj_002       | Snapshot | Completed                         | October 26, 2022, 17:56:44 (UTC+0900)   |
| <input type="checkbox"/> | kj_001       | Snapshot | Completed                         | October 26, 2022, 17:54:15 (UTC+0900)   |
| <input type="checkbox"/> | 800p3y9ggnnr | Snapshot | Completed                         | September 05, 2022, 14:13:19 (UTC+0900) |
| <input type="checkbox"/> | 9h4b3b0aztp4 | Snapshot | Completed                         | September 05, 2022, 13:59:51 (UTC+0900) |

Figure 96: Job Being Created

**ota\_job1** [Info](#)
[Refresh](#)
[Edit](#)
[Save as a job template](#)
[Cancel](#)
[Delete](#)

[Details](#)
[Job executions](#)
[Job document](#)
[Job targets](#)
[Tags](#)

**Job details**

|                                                                                                                                     |                                                                                                                                                               |                                                                                                                                                          |
|-------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Job name</b><br>ota_job1<br><br><b>ARN</b><br>arn:aws:iotap-northeast-1:432073875051:job/ota_job1<br><br><b>Description</b><br>- | <b>Last updated</b><br>October 27, 2022, 13:19:34 (UTC+0900)<br><br><b>Created</b><br>October 27, 2022, 13:19:29 (UTC+0900)<br><br><b>Status</b><br>Completed | <b>Devices to update</b><br>1 thing<br><br><b>Job run type</b><br>SNAPSHOT<br><br><b>Timeout configuration</b><br>-<br><br><b>Execution failure</b><br>- |
|-------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|

Figure 97: Successful Created Job

### 5.4 Execute OTA Update

When a job is created successfully, the device receives the job details as shown below.

```
[dpmAPMManager] DM_NEED_CONNECTION
DM_NEED_CONNECTION

[INFO] [DoorLockDemo] [aws_dpm_app_connect:2267] Establishing MQTT session with provisioned certificate...
recv timeout(=2000 ms) set OK (socket=0)
hostName = "alkzdt4nun8bnh-ats.iot.ap-northeast-1.amazonaws.com", flag to re-query (=0)
host IP from RTM = "54.178.218.11"
TCP connection OK to "alkzdt4nun8bnh-ats.iot.ap-northeast-1.amazonaws.com"
[INFO] [DoorLockDemo] [aws_dpm_app_connect:2317] Successfully established connection with provisioned
credentials.
[Make AWS-Thing-Name]
[NVRAM] AWS Thing name : [APP-DOORLOCK-1] (len=14)
[NVRAM] [APP-DOORLOCK-1/DeviceConnect] [APP-DOORLOCK-1/AppControl] [APP-DOORLOCK-1/DeviceControl]
[INFO] [DoorLockDemo] [aws_dpm_app_subscription:1939] subscription info: total(default:4, tried:4), OK(4)
current RTM user Timer ID = 5
current RTM temperature(str): 0.000000
current RTM battery(str): 0.000000
current RTM doorOpen state: "false"
current RTM doorOpenMode : 0
current RTM FOTAFlag: 1
current RTM FOTA url : "https://diasemi-ota-test3.s3.ap-northeast-2.amazonaws.com/"
[dpmAPMManager] DM_RTC_WAKEUP
DM_WAKEUP_TIMER (tid=5)

DEBUG: [aws_dpm_app_sensor_work:2104] read values from sensor if available

=====

recv timeout(=120 ms) set OK (socket=0)
[INFO] [DoorLockDemo] [aws_dpm_app_sensor_work:2162] publish (shadow sensor update) OK - payload:
{"state":{"reported":{"doorState":false,"temperature":4294967296.000000,"battery":4294967296.000000}}}}

last temperature: Not available
last battery: Not available
Sleep mode 3: KA timer interval(=1800 sec)
```

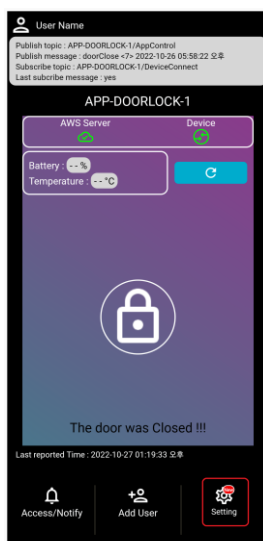
## DA16200 DA16600 Getting Started with AWS® IoT Core

```
DM_FINISH_DEVICE

recv timeout(=20 ms) set OK (socket=0)
[dpm_heartbeat_timer_register] RTC interval (=1780 secs), mode (=0)
>>> Start DPM Power-Down !!!
```

**Note**

- When a Job for an OTA update is created, users can see the URL of the S3 bucket accessed through JSON in the console. Also, the setting icon changes in the Mobile application. See the console message and [Figure 98](#)
- The temperature and battery value displayed as 4294967296 indicates that it is not available

**Figure 98: Successful Job for OTA Update in Mobile App**

The update is executed when the user clicks the **Update** button on the setting screen. The console and the Android application show the progress status during the OTA update. When the update is completed, the Thing restarts and in the Android device, the update notification disappears. See [Figure 99](#).

## DA16200 DA16600 Getting Started with AWS® IoT Core

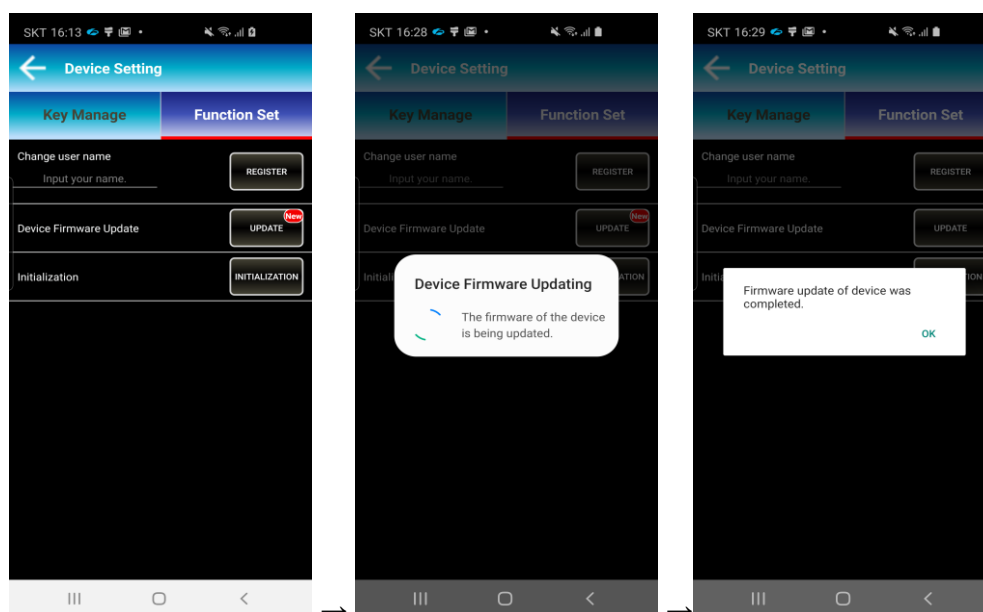


Figure 99: Execute OTA Update in Android App

The example below shows the console message when an update is being performed.

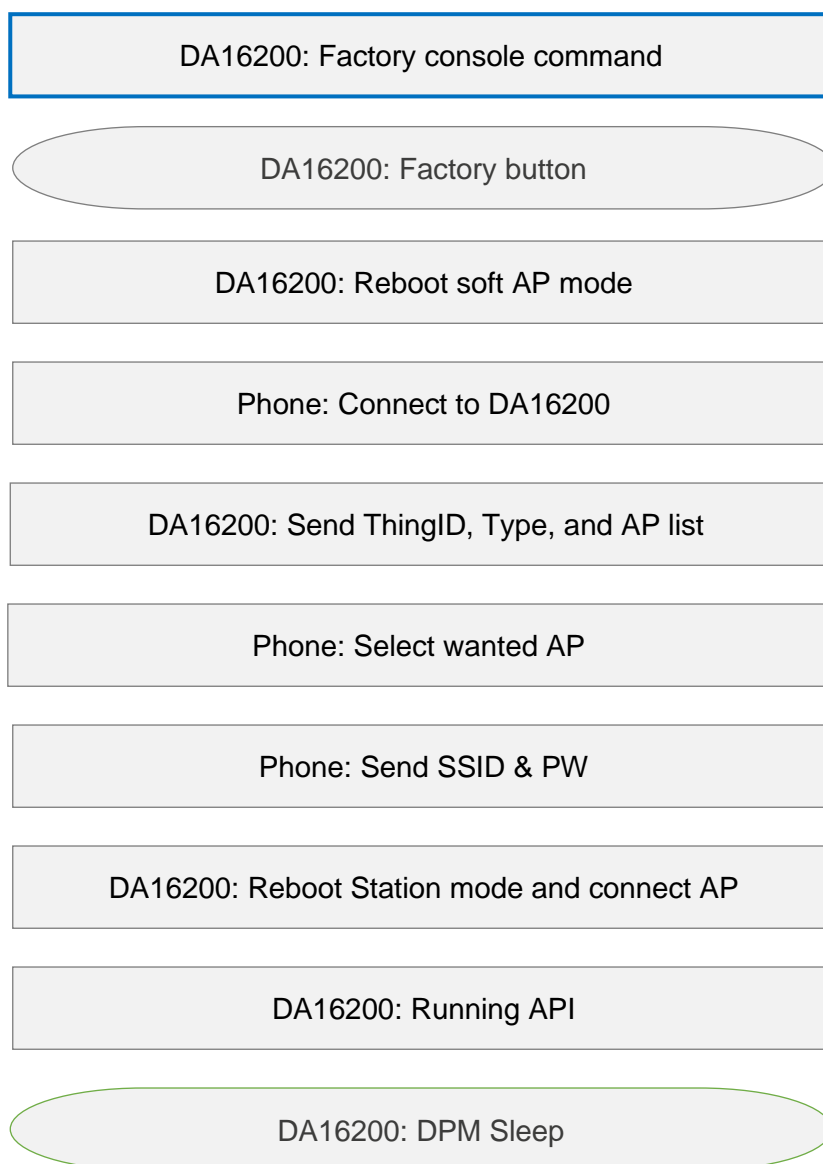
```
last user Timer ID = 5
last doorOpenFlag state: "false"
last FOTA Stat: 2
last FOTA Url: "https://diasemi-ota-test3.s3.ap-northeast-2.amazonaws.com/"
URL for updating: "https://diasemi-ota-test3.s3.ap-northeast-2.amazonaws.com/"
save URL info & reboot for OTA
Wakeup source is 0x0
...
...
...
DEBUG: [aws_ota_fw_update:3532] RTOS url https://diasemi-ota-test2.s3.ap-northeast-
2.amazonaws.com/DA16200_FRTOS-GEN01.img
>>> SNTP Server: pool.ntp.org (106.247.248.106)
>>> SNTP Time sync : 2022.10.26 - 08:56:58
> Server FW version : FRTOS-GEN01-01-56c232799-004457
>> HTTP(s) Client Downloading... 100 %(1202848/1202848 Bytes)
- OTA Update : <RTOS> Download - Success
DEBUG: [app_ota_fw_download_complete_notify:3375] RTOS download finish. (0x00)
- OTA: Renewing with new F/W
- OTA: RTOS
> Same Version : FRTOS-GEN01-01-56c232799-004457
>>> RTOS is updated and system reboots. (New boot_idx=0) !!!
DEBUG: [app_ota_fw_renew_notify:3497] Succeeded to replace with new FW.
- OTA: Reboot after 0 secs ...
Wakeup source is 0x0
[dpm_init_retmemory] DPM INIT CONFIGURATION(1)
```

## DA16200 DA16600 Getting Started with AWS® IoT Core

### Appendix A Provisioning

DA16200 supports a provisioning feature called Soft AP mode for an easy network configuration. Provisioning with the **mobile network data off** on your mobile phone and Wi-Fi turned on. When provisioning is complete, turn on your mobile data again. [Figure 100](#) shows the workflow of the provisioning process.

Press the **Factory Reset** button for about 5 seconds. Start the Android application and touch the **START** button to find the wanted AP.



**Figure 100: Provisioning Flow**

#### A.1 Android Application

```

System Mode : Soft-AP (1)

>>> DHCP Server Started
>>> Start DA16X Supplicant ...
>>> DA16x Supp Ver2.7 - 2022_03
>>> Add SoftAP Interface (softap1) ...
>>> MAC address (softap1) : d4:3d:39:11:5e:73
>>> softap1 interface add OK
>>> AP Operating Channel: 1(2412)

```

---

**DA16200 DA16600 Getting Started with AWS® IoT Core**

---

```
>>> Network Interface (wlan1) : UP
BSS Isolate Disabled

Soft-AP is Ready (d4:3d:39:11:5e:73)

=====
[APP-IOT Doorlock]
[aws_shadow_dpm_auto_start]
AWS_IOT on Station Mode for "APP-DOORLOCK-1"
=====

AWS_IOT AP Mode APP-DOORLOCK-1

=====
[Start Provisioning with TCP/TLS] .. Soft AP Mode
=====

[app_provision_switch_client_thread] Create...(status=0) [10]
[app_provision_TCP_server_thread] Create ...
[app_provision_TLS_server_thread] Create TLS...

>>> Start Provisioning Server (TLS) ...
Wait Accept (TLS)...

> Wi-Fi Scan request success.
(0) KT_GIGA_2G_505 / 3 / -25 / 2412
(1) TP-LINK_AECC / 3 / -40 / 2412

[app_provision_TCP_server_thread] socket().. status=1
Wait Accept...
```

## DA16200 DA16600 Getting Started with AWS® IoT Core

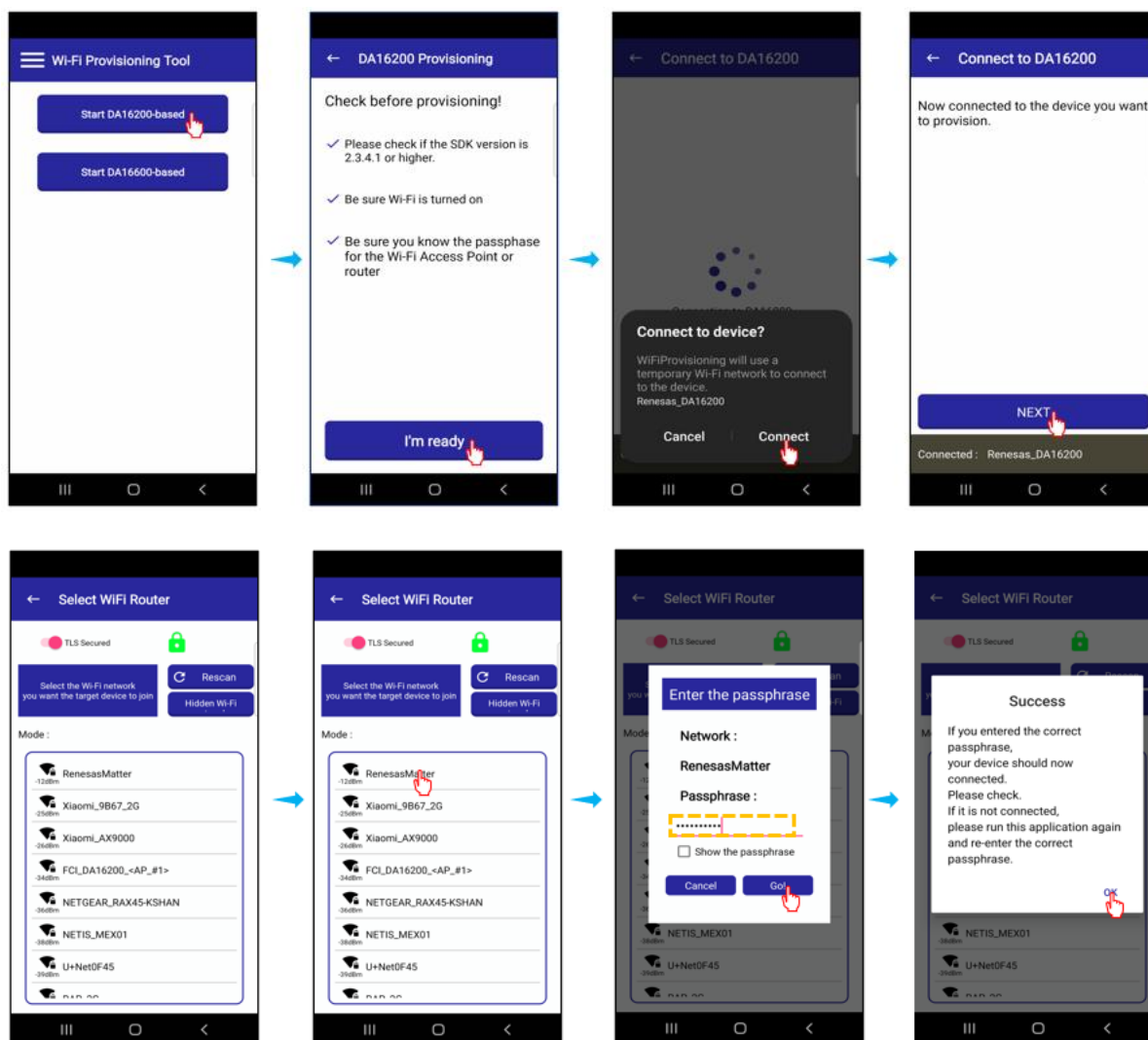


Figure 101: Provisioning from Mobile App

```
[dpmAPPManager] DM_NEED_CONNECTION
DM_NEED_CONNECTION

[INFO] [DoorLockDemo] [aws_dpm_app_connect:2267] Establishing MQTT session with provisioned certificate...
recv timeout(=2000 ms) set OK (socket=0)
hostName = "alkzdt4nun8bnh-ats.iot.ap-northeast-1.amazonaws.com", flag to re-query (=0)
host IP = "52.69.14.255"
TCP connection OK to "alkzdt4nun8bnh-ats.iot.ap-northeast-1.amazonaws.com"
recv timeout(=120 ms) set OK (socket=0)
[INFO] [DoorLockDemo] [aws_dpm_app_connect:2317] Successfully established connection with provisioned
credentials.
[Make AWS-Thing-Name]
[NVRAM] AWS Thing name : [APP-DOORLOCK-1] (len=14)
[NVRAM] [APP-DOORLOCK-1/DeviceConnect] [APP-DOORLOCK-1/AppControl] [APP-DOORLOCK-1/DeviceControl]
[INFO] [DoorLockDemo] [aws_dpm_app_subscription:1939] subscription info: total(default:4, tried:4), OK(4)
current RIM user Timer ID = 0
current RIM temperature(str): 0.000000
current RIM battery(str): 0.000000
current RIM doorOpen state: "false"
current RIM doorOpenMode : 0
current RIM FOTAflag: 0
current RIM FOTA url : ""
[dpmAPPManager] DM_BOOT_WAKEUP
DM_WAKEUP_BOOT

[INFO] [DoorLockDemo] [connectionReadyInform:1598] publish (command response) OK - payload: "yes"
```

## DA16200 DA16600 Getting Started with AWS® IoT Core

```
[closeControl]

[INFO] [DoorLockDemo] [aws_dpm_app_door work:2030] publish (shadow doorlock update) OK - payload:
{"state":{"reported":{"doorState":false,"openMethod":"app","doorStateChange":1,"doorOpenMode":0,"OTAupdate":0
,"OTAresult":"OTA_UNKNOWN"}}}

last user Timer ID = 0
last doorOpenFlag state: "false"
last FOTA Stat: 0
last FOTA Url: ""

Sleep mode 3: KA timer interval(=1800 sec)

DM_FINISH_DEVICE

rcv timeout(=20 ms) set OK (socket=0)
[dpm_heartbeat_timer_register] RTC interval (=1780 secs), mode (=0)
>>> Start DPM Power-Down !!!
```

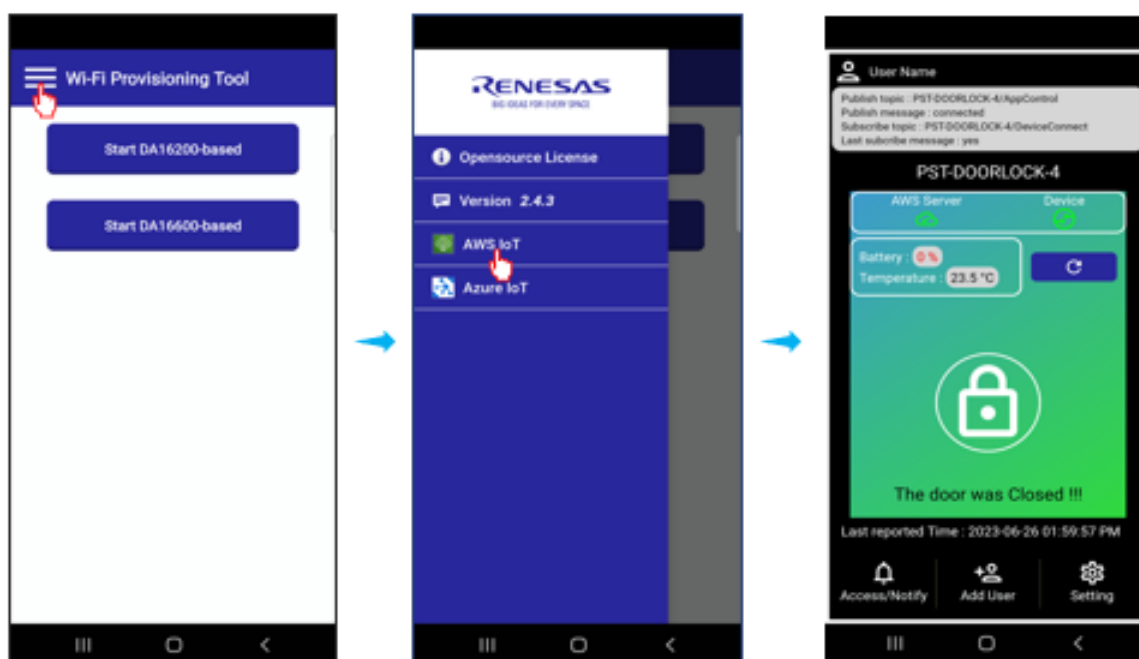


Figure 102: Running AWS IoT Application from Mobile App

## Appendix B AT Commands for AWS IoT

### B.1 Operating Modes

There are three operating modes:

- Setting Mode for features configuration
- Provisioning Mode for network connection
- Communicating Mode for running

#### B.1.1 Setting Mode

After uploading the image and rebooting, the DA16200/DA16600 enters the setting mode. In this mode, all AWS IoT settings can be configured using the SET command and a specific topic can be



## DA16200 DA16600 Getting Started with AWS® IoT Core

configured using the CFG command. For proper operation of AWS IoT, the TLS certificate keys must be set. All configuration data is stored before calling the factory reset command (see [Figure 103](#)).

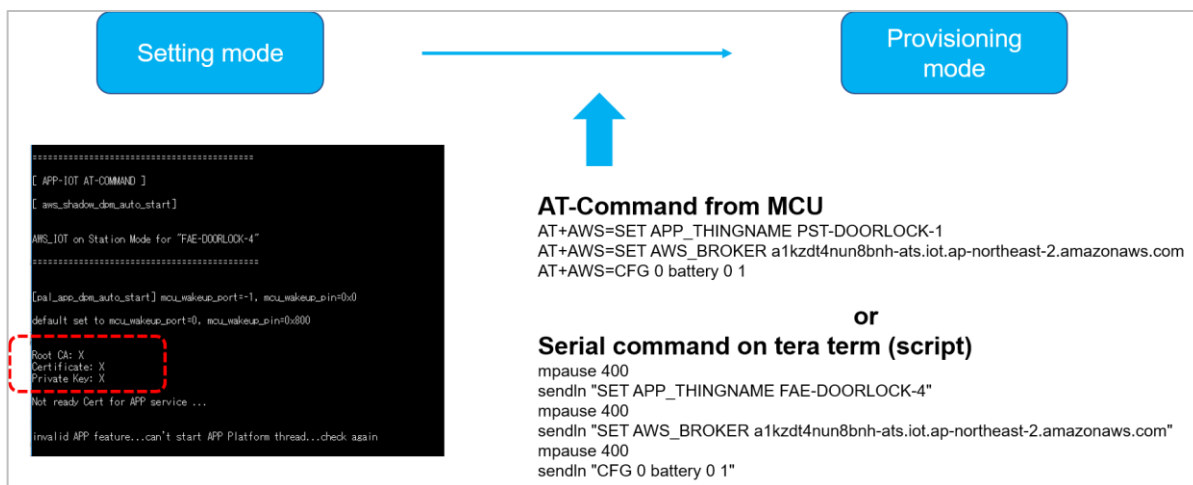


Figure 103: Setting Mode

### B.1.2 Provisioning Mode

In provisioning mode, the DA16200/DA16600 can be provisioned using an Android or iOS device. During provisioning, the MCU only receives a report on the provisioning status. When provisioning is complete, the DA16200/DA16600 will enter communication mode automatically after rebooting (see [Figure 104](#)).

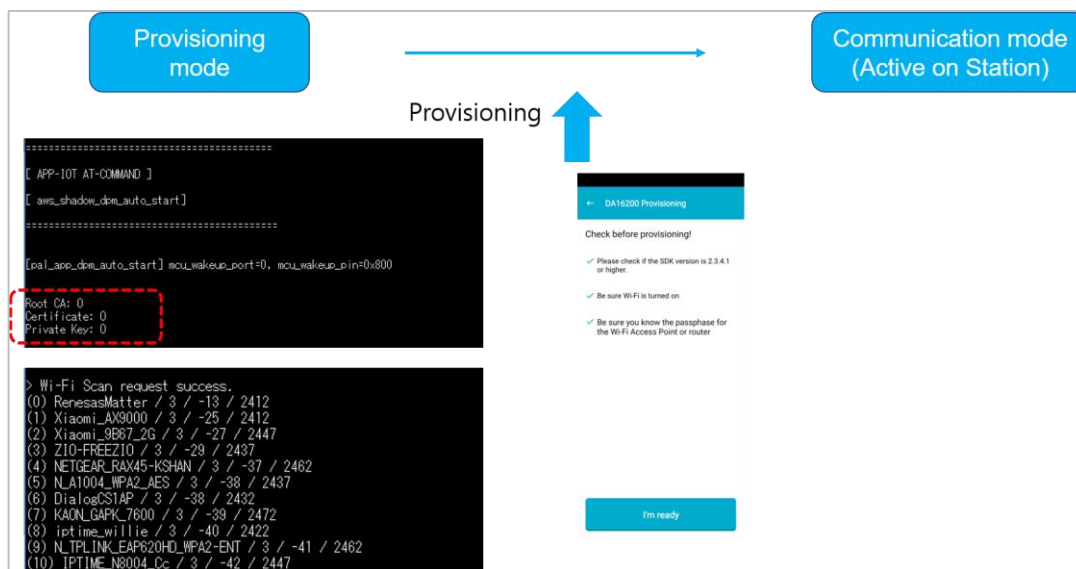


Figure 104: Provisioning Mode

### B.1.3 Communication Mode

The DA16200/DA16600 Communication Mode is used by the MCU to communicate (send and receive) topic values with an AWS server (See [Figure 105](#)).

DA16200 DA16600 Getting Started with AWS® IoT Core

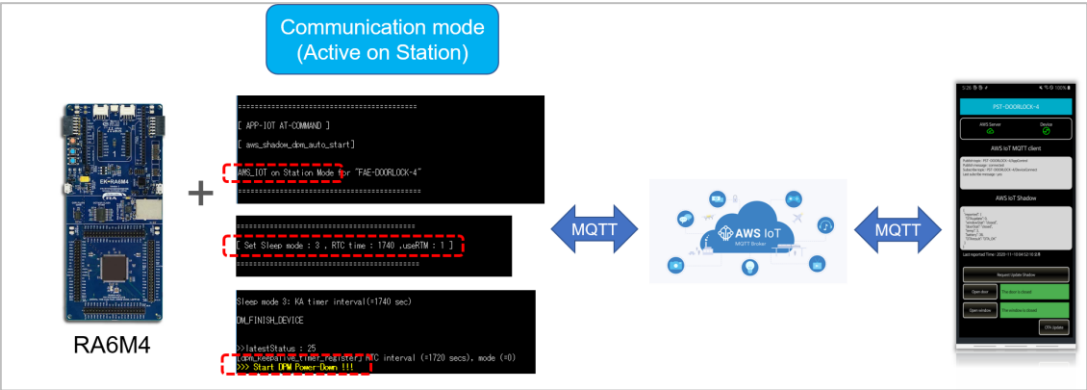


Figure 105: Communication Mode

B.2 Configuring Topic to Publish, Subscribe and Shadow

B.2.1 Configure Topics

- Topics are configured as shown in Table 5
  - The MCU and Mobile App should be configured based on the topics shown in Table 5
  - The MCU pushes the topics in Table 5 to the DA16200/DA16600 using AT command
- The DA16200 facilitates the communication between the MCU and phone as shown in Figure 106

Table 5: Configuration of Topics

| Number | Name        | Value Type | CMD Type     | Value                 |
|--------|-------------|------------|--------------|-----------------------|
| 0      | app_door    | 1: String  | 2: Subscribe | "open"/"close"        |
| 1      | mcu_door    | 1: String  | 0: Publish   | "opened"/"closed"     |
| 2      | battery     | 0: Integer | 1: Shadow    | Battery value (0~100) |
| 3      | temperature | 2: Float   | 1: Shadow    | Temperature value     |
| 4      | doorStat    | 1: String  | 1: Shadow    | "opened"/"closed"     |
| 5      | windowStat  | 1: String  | 1: Shadow    | "opened"/"closed"     |

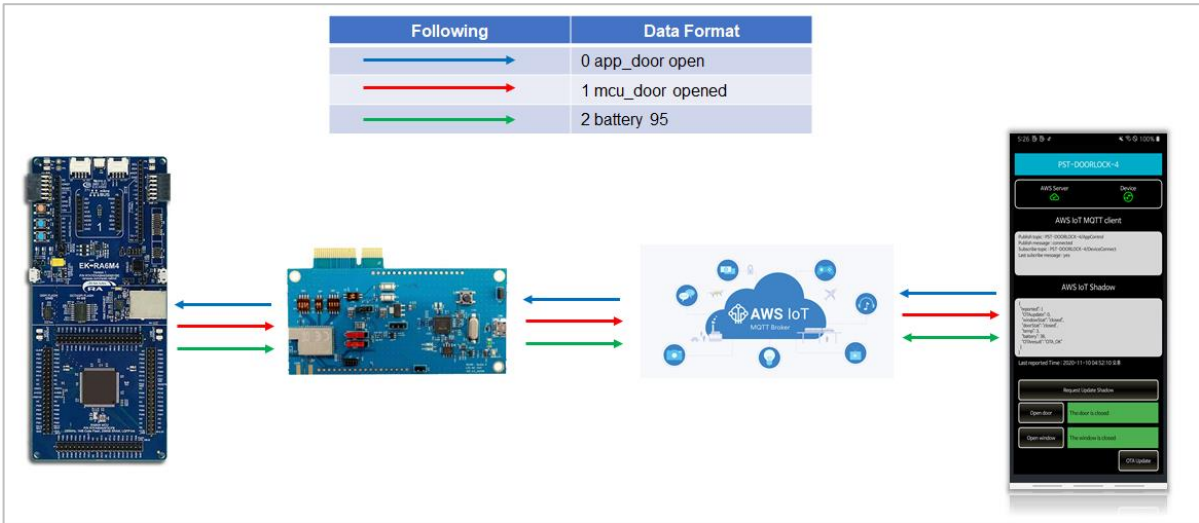


Figure 106: Communication between MCU and Phone

## DA16200 DA16600 Getting Started with AWS® IoT Core

### B.3 AT Command List

#### B.3.1 Basic Set

Table 6: Basic Set of MCU to DA16200/DA16600

| Head                                                                                                                        | Main | Sub            | Parameters                                                                                                                                                                                                                        |
|-----------------------------------------------------------------------------------------------------------------------------|------|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AT+AWS=SET                                                                                                                  |      | APP_THINGNAME  | Set the device thing name<br>Used to choose a device by its thing name during provisioning                                                                                                                                        |
|                                                                                                                             |      | AWS_BROKER     | Set the broker address                                                                                                                                                                                                            |
|                                                                                                                             |      | APP_LPORT      | Set the local port                                                                                                                                                                                                                |
|                                                                                                                             |      | APP_SUBTOPIC   | Set subscriber topic name, and the default is “/AppControl”                                                                                                                                                                       |
|                                                                                                                             |      | APP_PUBTOPIC   | Set subs topic name, and the default is “/DeviceControl”                                                                                                                                                                          |
|                                                                                                                             |      | SLEEP_MODE     | Set sleep mode<br>1 – not connected sleep. DA16200/DA16600 will wake up only by RTC_PWR_KEY.<br>2 – not connected sleep. DA16200/DA16600 will wake up by RTC.<br>3 - connected sleep. The connection is retained even during DPM. |
|                                                                                                                             |      | USE_DPM        | Define the operation of sleep mode 3<br>0 - no DPM. Used during debug<br>1 - DPM mode                                                                                                                                             |
|                                                                                                                             |      | RTC_TIME       | Set the wake-up time for Sleep mode 2                                                                                                                                                                                             |
|                                                                                                                             |      | DPM_KEEP_ALIVE | Set the keep-alive time between the IoT device and the AP<br>Default value is 30*1000 microseconds                                                                                                                                |
|                                                                                                                             |      | USE_WAKE_UP    | Set the wake-up time for full-boot mode<br>Default value is set to 0 (0 = unused)                                                                                                                                                 |
|                                                                                                                             |      | TIM_WAKE_UP    | Set the period to check a beacon frame from the AP<br>Default value is set to 10                                                                                                                                                  |
|                                                                                                                             |      | AWS_USE_FP     | Not used command<br>0 – Default value<br>1 – Not in use                                                                                                                                                                           |
| EX) AT+AWS=SET APP_THINGNAME AssignedThingName<br>AT+AWS=SET AWS_BROKER a1kzdt4nun8bnh-ats.iot.ap-northeast-2.amazonaws.com |      |                |                                                                                                                                                                                                                                   |

#### B.3.2 TLS Certificate

Table 7: TLS from MCU to DA16200/DA16600

| Start Code | Sub Code | Type                                                                                                            | End Code |
|------------|----------|-----------------------------------------------------------------------------------------------------------------|----------|
| \x1b       | C0,      | Root CA<br>Self-Signed, well known<br>Has root certificate public key<br>Signed by root certificate private key | \x03     |

## DA16200 DA16600 Getting Started with AWS® IoT Core

| Start Code                                                                                                                           | Sub Code | Type                                                                                                                                     | End Code |
|--------------------------------------------------------------------------------------------------------------------------------------|----------|------------------------------------------------------------------------------------------------------------------------------------------|----------|
|                                                                                                                                      | C1,      | Certificate key<br>Has own public key<br>Signed by root certificate private key<br>Use root certificate public key to prove authenticity |          |
|                                                                                                                                      | C2,      | Private key<br>Has own public key<br>Signed by certificate private key<br>Use certificate 1 public key to prove authenticity             |          |
| EX) send "\x1b" over UART<br>send "C0,-----BEGIN CERTIFICATE-----\n" "MIIDQTCCAimgAwIBAgITBmyfz5m/jAo ..... over UART<br>send "\x03" |          |                                                                                                                                          |          |

## B.3.3 PIN MUX

Table 8: PIN MUX from MCU to DA16200/DA16600

| Head    | Main                | Sub                    | Parameters  |                              |                       |
|---------|---------------------|------------------------|-------------|------------------------------|-----------------------|
| AT+AWS= | SET                 | NV_PIN_AMUX            | AMUX_UART1d | 4                            | /* UART1(RXD, TXD) */ |
|         |                     |                        | AMUX_GPIO   | 9                            | /* GPIOA [1:0] */     |
|         |                     | NV_PIN_BMUX            | BMUX_UART1d | 4                            | /* UART1(RXD, TXD) */ |
|         |                     |                        | BMUX_GPIO   | 8                            | /* GPIOA [3:2] */     |
|         |                     | NV_PIN_CMUX            | CMUX_UART1d | 6                            | /* UART1(RXD, TXD) */ |
|         |                     |                        | CMUX_GPIO   | 8                            | /* GPIOA [5:4] */     |
|         |                     | NV_PIN_DMUX            | DMUX_UART1d | 4                            | /* UART1(RXD, TXD) */ |
|         |                     |                        | DMUX_GPIO   | 8                            | /* GPIOA [7:6] */     |
|         |                     | NV_PIN_EMUX            | EMUX_GPIO   | 8                            | /* GPIOA [9:8] */     |
|         |                     | NV_PIN_FMUX            | FMUX_GPIO   | 6                            | /* GPIOA [11:10] */   |
|         | NV_PIN_UMUX         | UMUX_GPIO              | 2           | /* GPIOC [8:6] */            |                       |
|         | APP_MCU_WKAEUP_PORT | GPIO_UNIT_A            | 0           |                              |                       |
|         |                     | GPIO_UNIT_C            | 2           | /*Support only GPIO 6,7,8 */ |                       |
|         | APP_MCU_WKAEUP_PIN  | GPIO_PIN0 ~ GPIO_PIN11 |             |                              |                       |
|         | UART_CFG            | [baud-rate]            |             |                              |                       |

Note: Default pin mux is BMUX  
Ex) use GPIOA2 and GPIOA3 for UART1, and GPIOA9 for MCU wakeup  
AT+AWS=SET NV\_PIN\_BMUX BMUX\_UART1d  
AT+AWS=SET NV\_PIN\_EMUX EMUX\_GPIO  
AT+AWS=SET APP\_MCU\_WKAEUP\_PORT GPIO\_UNIT\_A  
AT+AWS=SET APP\_MCU\_WKAEUP\_PIN GPIO\_PIN9

## DA16200 DA16600 Getting Started with AWS® IoT Core

### B.3.4 Configure Data as Topics

**Table 9: Configuration Data from MCU to DA16200/DA16600**

| Head                                                                                    | Main | Sub                                         | Parameters                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-----------------------------------------------------------------------------------------|------|---------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AT+AWS=                                                                                 | CFG  | [number] [name] [value-type]<br>[MQTT-type] | <ul style="list-style-type: none"> <li>number:<br/>index to identify the saved topic<br/>Increase by 1 when setting a new topic<br/>Max value is 10 (total supported topics is 10)</li> <li>name:<br/>String specifying the topic name</li> <li>value-type<br/>0 - Integer type<br/>1 - String type<br/>2 - Float type</li> <li>MQTT-type<br/>0 - Publish: The prompt command is used to send a value from the MCU to the phone. For example, door state = true/false<br/>1 - Shadow: The value is sent to the device twin and will be updated on the phone the next time it is connected.<br/>2 - Subscribe: The prompt command is used to send a value from the phone to the MCU. For example, door open command.</li> </ul> |
| Ex) AT+AWS=CFG 0 doorStat 1 1<br>AT+AWS=CFG 1 battery 2 1<br>AT+AWS=CFG 2 door_open 0 2 |      |                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |

### B.3.5 Command – MCU to DA16200/DA16600

**Table 10: Command of MCU to DA16200/DA16600**

| Head                         | Main | Sub           | Description                                                                                                                                                              |
|------------------------------|------|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AT+AWS=                      | CMD  | FACTORY_RESET | Reset the AWS IoT configuration to the factory default. All values stored in NVRAM are cleared<br>Use the “SET” and “CFG” commands to set the AWS IoT configuration      |
|                              |      | RESET_TO_AP   | Switch to AP mode keeping the values set in NVRAM<br>The previous values in NVRAM will be kept                                                                           |
|                              |      | GET_STATUS    | Get the current AWS IoT status<br>The MCU can read the current status from the DA16200/DA16600 at any time                                                               |
|                              |      | RESTART       | Reboot the device keeping the current mode and status                                                                                                                    |
|                              |      | MCU_DATA      | Used by the MCU to set a CFG parameter in the DA16200/DA16600. The value must be the same format as defined by the CFG setting<br>Parameters:<br>[number] [name] [value] |
| Ex) AT+AZU=CMD FACTORY_RESET |      |               |                                                                                                                                                                          |

## DA16200 DA16600 Getting Started with AWS® IoT Core

| Head                                  | Main | Sub | Description |
|---------------------------------------|------|-----|-------------|
| AT+AZU=CMD MCU_DATA 1 mcu_door opened |      |     |             |

### B.3.6 Command – DA16200/DA16600 to MCU

**Table 11: Command of DA16200/DA16600 to MCU**

| Head                                                                     | Main        | Parameters                 | Description                                                                                                                                                                                                                                                                      |
|--------------------------------------------------------------------------|-------------|----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| +AWSIOT                                                                  | SERVER_DATA | [number] [name]<br>[value] | Used by the DA16200/DA16600 to set a CFG parameter in the MCU. The value must be the same format as defined by the CFG setting.                                                                                                                                                  |
| +AWSIOT                                                                  | CMD_TO_MCU  | update                     | Used by the DA16200/DA16600 to request the status of devices such as sensors, batteries, and doors from the MCU. The DA16200/DA16600 maintains the values obtained from the MCU and forwards them when requested by an external phone app or by an MQTT ping-pong wake-up event. |
| Ex) +AWSIOT SERVER_DATA 0 door_control open<br>+AWSIOT CMD_TO_MCU update |             |                            |                                                                                                                                                                                                                                                                                  |

### B.3.7 DA16200/DA16600 Status – DA16200/DA16600 to MCU

**Table 12: Status from DA16200/DA16600 to MCU**

| Status                | Value | Parameters                                                                                                                                                                                   |
|-----------------------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| IDLE                  | -1    | Initial state of AWS-IoT application<br>Sent when a system error occurs. For example, network connection failure                                                                             |
| Done factory reset    | 0     | Sent after completes factory reset by "CMD FACTORY_RESET"                                                                                                                                    |
| Boot Ready            | 1     | Sent when entering AWS-IoT application mode                                                                                                                                                  |
| Need configuration    | 5     | Sent if there is no setting<br>MCU should set and configure with the SET and CFG command                                                                                                     |
| Start AP mode         | 10    | Sent when being started to AP mode<br>Need to process provisioning with Phone                                                                                                                |
| Network OK            | 15    | Sent when it is OK to connect AP without problem                                                                                                                                             |
| Network fail          | 16    | Sent when it fails to connect AP with any problem<br>Normally, it will happen during provisioning failure by the wrong SSID or PW<br>Need to go to AP mode by MCU send "RESET_TO_AP" command |
| Start STA             | 20    | Not defined yet                                                                                                                                                                              |
| Done STA              | 25    | Sent when entering sleep mode for DPM                                                                                                                                                        |
| MCUOTA                | 30    | Sent when MCU OTA starts processing                                                                                                                                                          |
| EX) +AWSIOT STATUS 15 |       |                                                                                                                                                                                              |

## Appendix C Troubleshooting

### C.1 Operational Issue

When UI buttons are not visible or not showing up properly while using the mobile app, try to uninstall and install it again. The first time running the mobile app after reinstalling it, make sure that the app can access the location of the device as described in Test Provisioning on Android/iPhone sections of [\[4\]](#).

## Revision History

| Revision | Date          | Description                                                                                                                                                                                                                                                   |
|----------|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1.5      | Jan. 26, 2024 | Added Troubleshooting section                                                                                                                                                                                                                                 |
| 1.4      | Nov. 30, 2023 | Merged documents: <ul style="list-style-type: none"><li>• UM-WI-016 DA16200 Door Lock Application Using AWS IoT</li><li>• UM-WI-017 DA16200 AWS IoT Server Setup</li><li>• UM-WI-038 DA16200 DA16600 Getting Started with AWS IoT Using AT Commands</li></ul> |
| 1.3      | Aug. 18, 2023 | <ul style="list-style-type: none"><li>• Changed IDE to e2studio</li><li>• Editorial update</li></ul>                                                                                                                                                          |
| 1.2      | Dec. 01, 2022 | Edited as direct link of documents                                                                                                                                                                                                                            |
| 1.1      | Nov. 04, 2022 | Modify hyperlink of the documents                                                                                                                                                                                                                             |
| 1.0      | Oct. 13, 2022 | Initial version.                                                                                                                                                                                                                                              |



---

**DA16200 DA16600 Getting Started with AWS® IoT Core****Status Definitions**

| Status                  | Definition                                                                                                                   |
|-------------------------|------------------------------------------------------------------------------------------------------------------------------|
| DRAFT                   | The content of this document is under review and subject to formal approval, which may result in modifications or additions. |
| APPROVED<br>or unmarked | The content of this document has been approved for publication.                                                              |

**RoHS Compliance**

Renesas Electronics' suppliers certify that its products are in compliance with the requirements of Directive 2011/65/EU of the European Parliament on the restriction of the use of certain hazardous substances in electrical and electronic equipment. RoHS certificates from our suppliers are available on request.

---

## DA16200 DA16600 Getting Started with AWS® IoT Core

### Important Notice and Disclaimer

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES ("RENESAS") PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES "AS IS" AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers skilled in the art designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only for development of an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising out of your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

## Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu  
Koto-ku, Tokyo 135-0061, Japan  
[www.renesas.com](http://www.renesas.com)

### Contact Information

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

<https://www.renesas.com/contact/>

### Trademarks

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