User Manual

AWS IoT

This guide walks through the steps to use AWS IoT and create Thing and Shadow in EasyBuilder Pro.

20220823

Table of Contents

1. Overview of AWS IoT

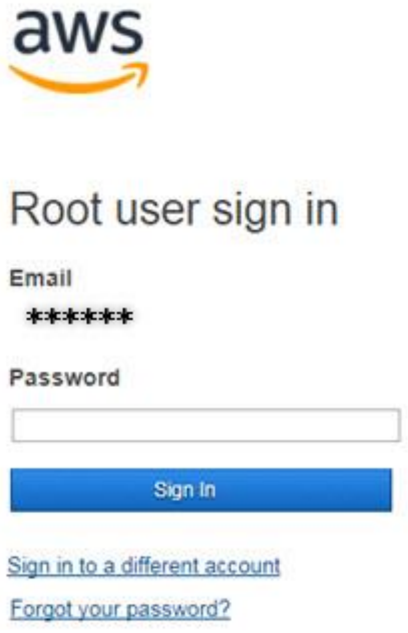
AWS (Amazon Web Service) is a cloud platform now widely used on the market, and AWS IoT (Internet of Things) supports MQTT protocol. Observing the market trend, from EasyBuilder Pro V6.00.01, Weintek has adopted AWS IoT service and integrated it with the MQTT feature released earlier. Apart from using AWS IoT as a broker in the publish-subscribe mode, users can also create Thing and Shadow offered by AWS IoT to make the most of MQTT.

This manual walks you through the steps to host MQTT server, configure EasyBuilder Pro, and create Things.

2. Hosting MQTT Server

AWS is a cloud platform; therefore, the settings are all configured on the web, please sign up in Amazon website before hosting an MQTT server.

1. Visit Amazon Web Service website at https://aws.amazon.com
2. Sign up. (left: individual account, right: business account)

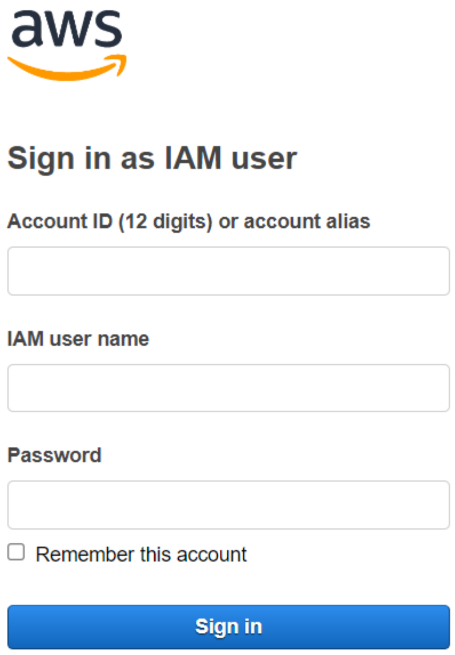


Root user sign in

Email

******

Password



Sign in as IAM user

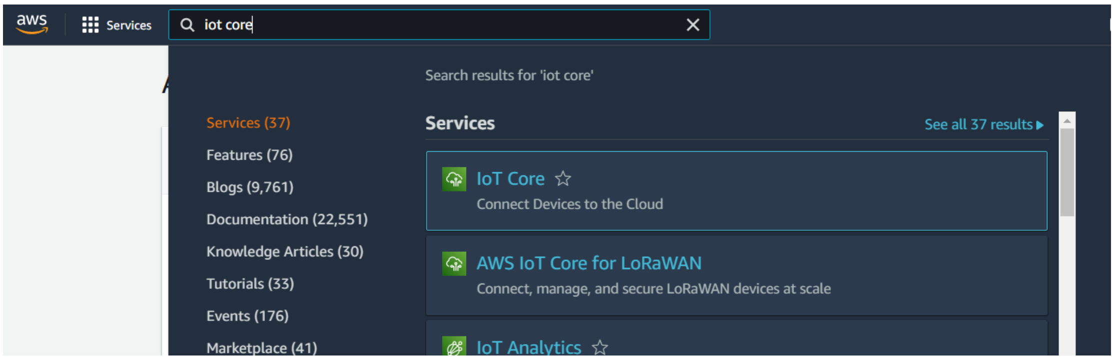
Account ID (12 digits) or account alias

IAM user name

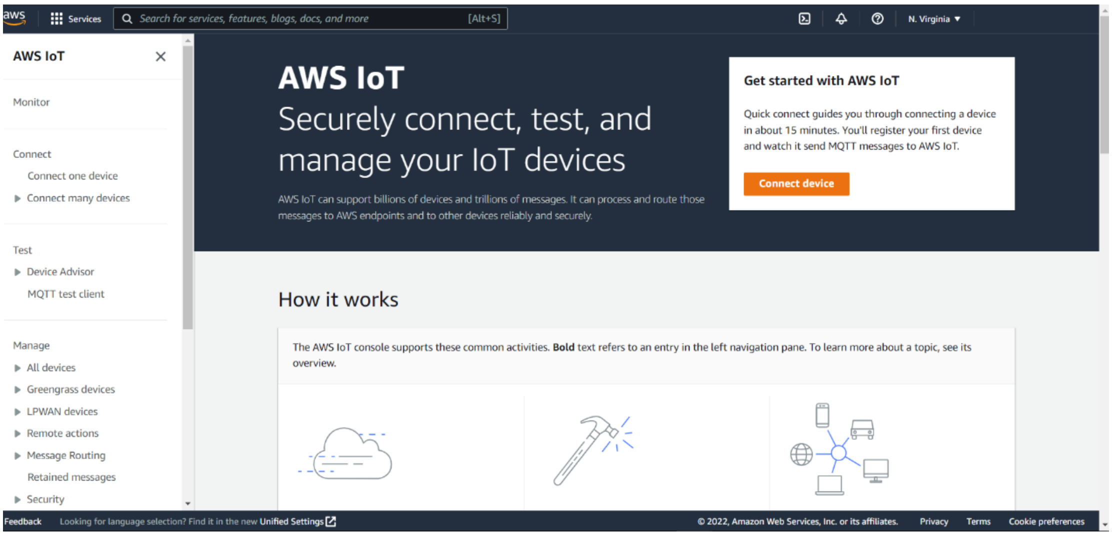
Password

□ Remember this account

3. After sign in, browse for IoT Core.



4. On the AWS IoT page, Policy and Certificate can be created



AWS IoT

Securely connect, test, and manage your IoT devices

AWS IoT can support billions of devices and trillions of messages. It can process and route those messages to AWS endpoints and to other devices reliably and securely.

Get started with AWS IoT

Quick connect guides you through connecting a device in about 15 minutes. You'll register your first device and watch it send MQTT messages to AWS IoT.

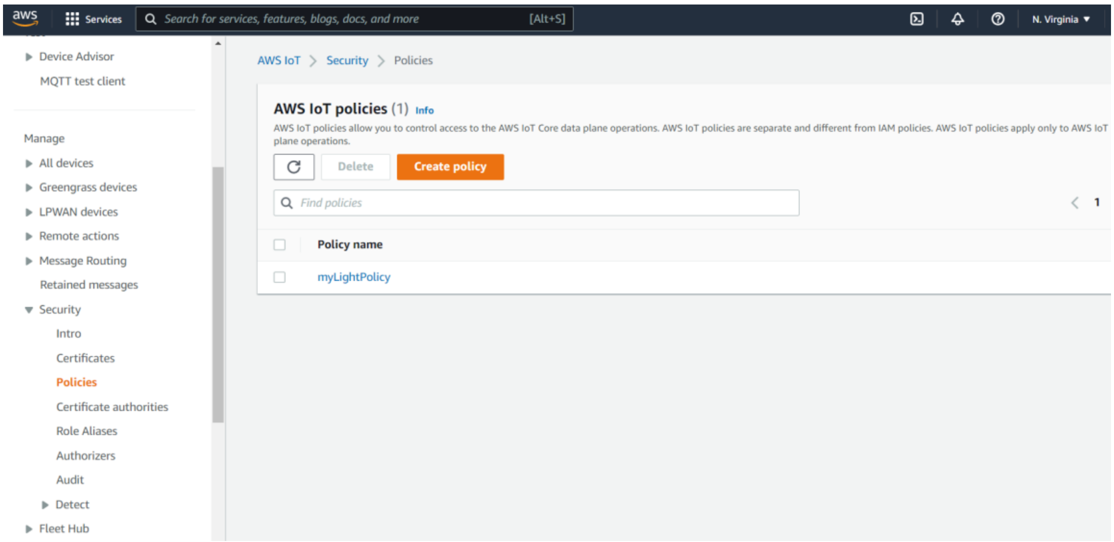
How it works

The AWS IoT console supports these common activities. **Bold** text refers to an entry in the left navigation pane. To learn more about a topic, see its overview.

Looking for language selection? Find it in the new Unified Settings

© 2022, Amazon Web Services, Inc. or its affiliates.

5. Open [Manage] » [Security] » [Policies] and then click [Create policy].

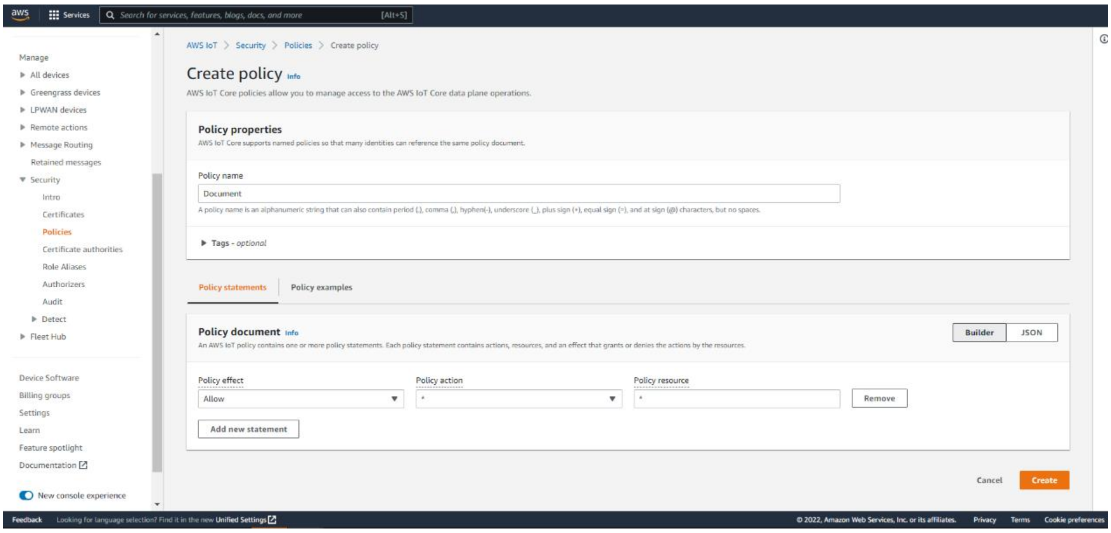


AWS IoT policies (1)

AWS IoT policies allow you to control access to the AWS IoT Core data plane operations. AWS IoT policies are separate and different from IAM policies. AWS IoT policies apply only to AWS IoT plane operations.

□	Policy name
□	myLightPolicy

6. This page is for defining actions that can be performed by a resource. You may use the settings in the screenshot below or set your own. Click [Create] when finish.



Create policy

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: Document

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.

Policy document

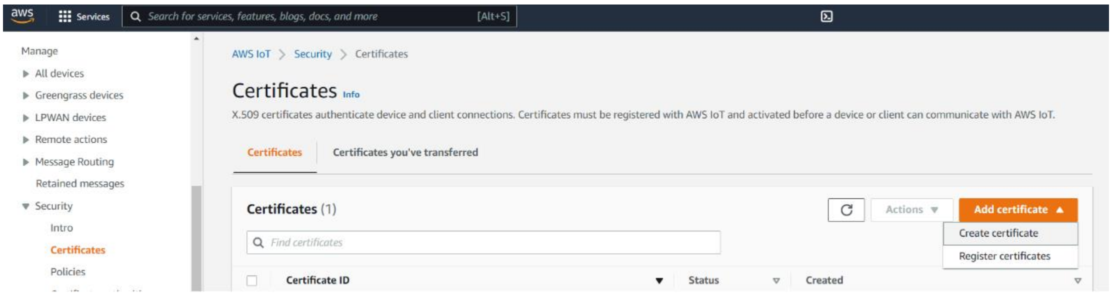
An AWS IoT policy contains one or more policy statements. Each policy statement contains actions, resources, and an effect that grants or denies the actions by the resources.

Policy effect: Allow

Policy action: *

Policy resource: *

7. Click [Security] » [Certificates] and then click [Add certificate] » [Create certificate].

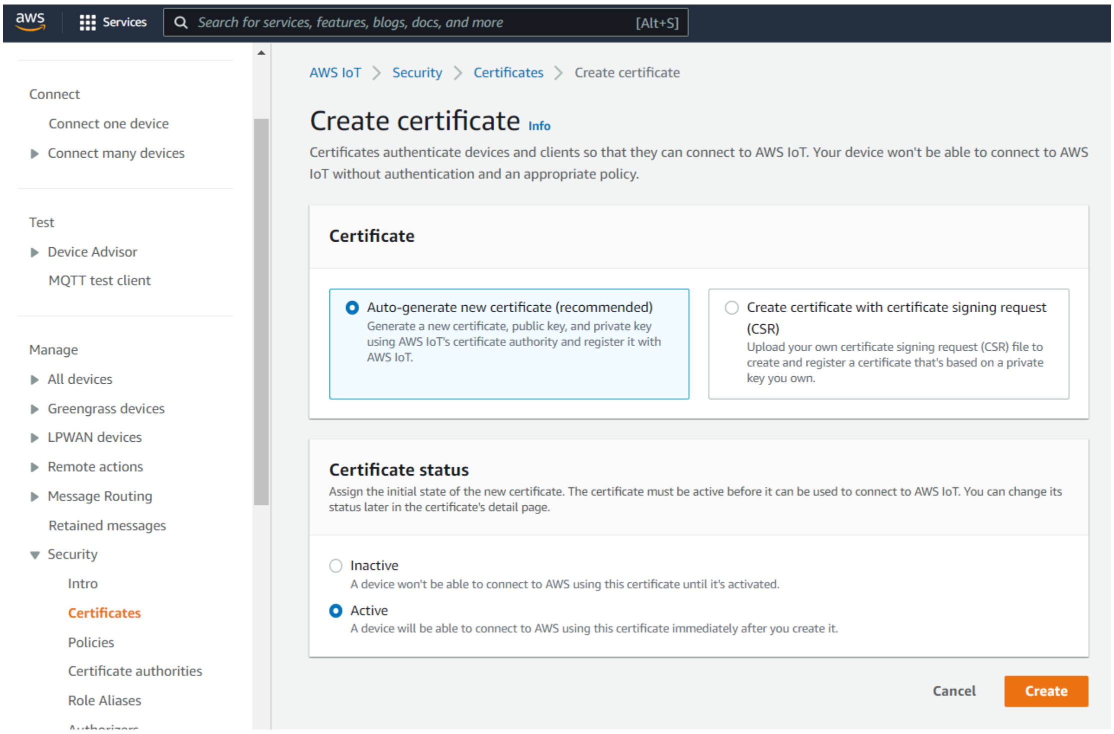


Certificates

X.509 certificates authenticate device and client connections. Certificates must be registered with AWS IoT and activated before a device or client can communicate with AWS IoT.

Certificates (1)

8. Select [Auto-generate new certificate (recommended)], set Certificate Status to [Active], and then click [Create].



Create certificate

Certificates authenticate devices and clients so that they can connect to AWS IoT. Your device won't be able to connect to AWS IoT without authentication and an appropriate policy.

Certificate

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

○ Create certificate with certificate signing request (CSR)
Upload your own certificate signing request (CSR) file to create and register a certificate that's based on a private key you own.

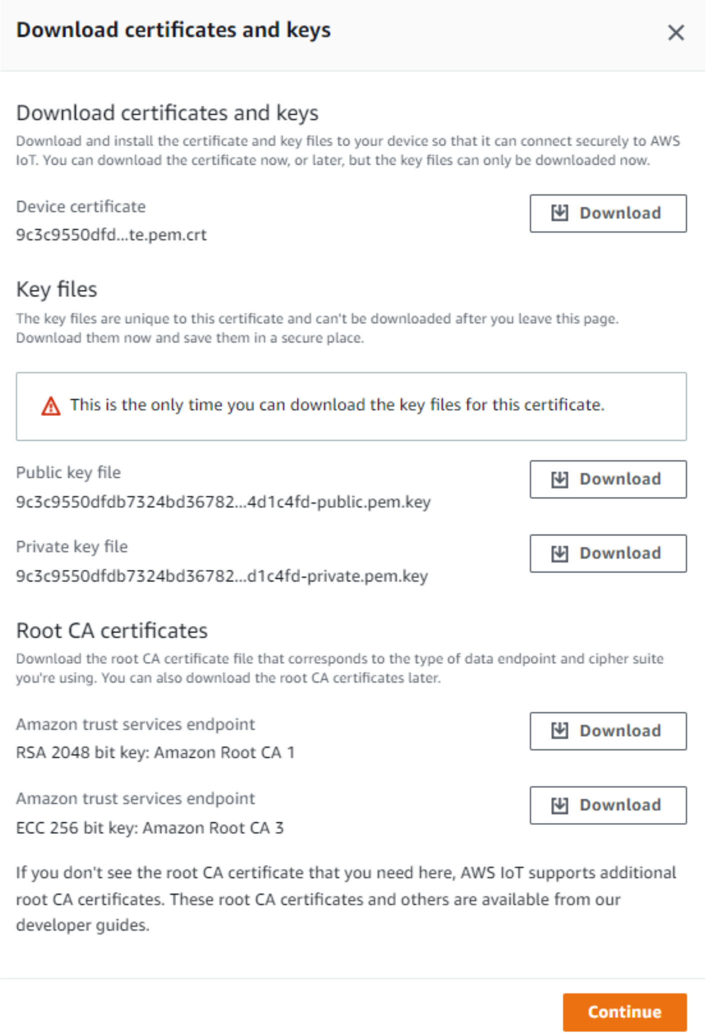
Certificate status

Assign the initial state of the new certificate. The certificate must be active before it can be used to connect to AWS IoT. You can change its status later in the certificate's detail page.

○ Inactive
A device won't be able to connect to AWS using this certificate until it's activated.

● Active
A device will be able to connect to AWS using this certificate immediately after you create it.

9. Download and save these files: Device Certificate, Public Key File, Private Key File, and RSA 2048 bit key: Amazon Root CA 1.



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
9c3c9550dfd...te.pem.crt

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
9c3c9550dfdb7324bd36782...4d1c4fd-public.pem.key

Private key file
9c3c9550dfdb7324bd36782...d1c4fd-private.pem.key

Root CA certificates

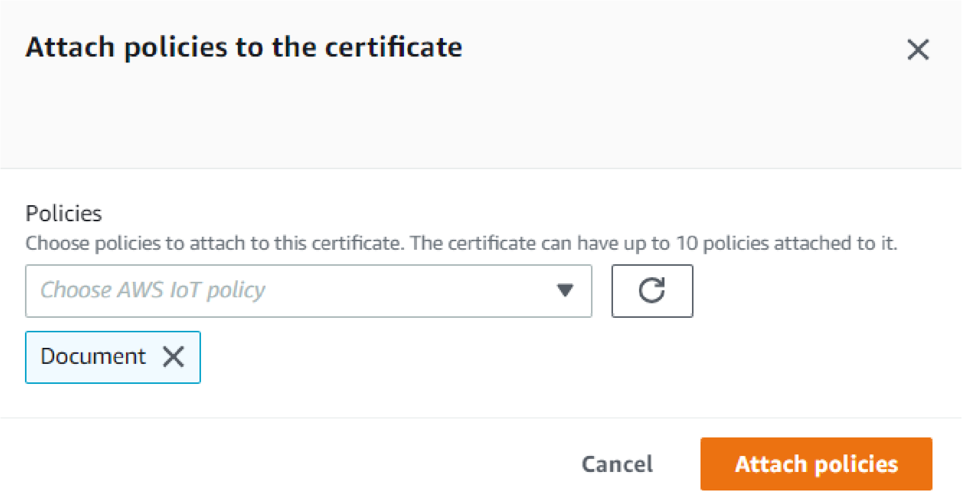
Download the root CA certificate file that corresponds to the type of data endpoint and cipher suite you're using. You can also download the root CA certificates later.

Amazon trust services endpoint
RSA 2048 bit key: Amazon Root CA 1

Amazon trust services endpoint
ECC 256 bit key: Amazon Root CA 3

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.

10. Click the certificate created previously and click [Attach policies] under Policies. In the window that follows, select the Policy created previously and then click [Attach policies].

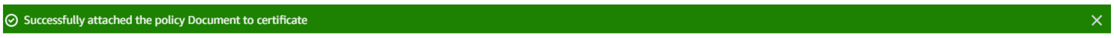


Attach policies to the certificate

Policies

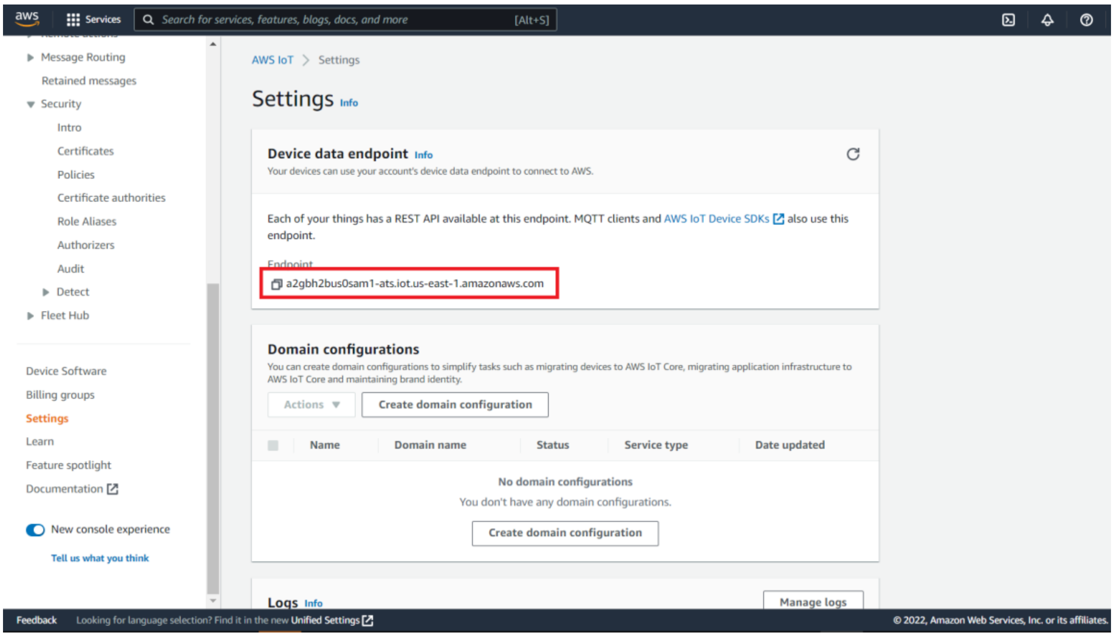
Choose policies to attach to this certificate. The certificate can have up to 10 policies attached to it.

11. Security setting is done successfully when the following box shows.



Successfully attached the policy Document to certificate

12. Click [Settings], the URL marked in red frame below is the domain name of AWS IoT server, and will be used when setting MQTT in EasyBuilder Pro, please remember it.
    AWS is gradually replacing servers using Symantec CA with Amazon Trust Service. As a result, please check whether the domain name contains "-ats", for example: a2xxxxxxx-ats.iot.xxxxxx.amazon.aws.com.
    Amazon Root CA 1 certificate created in step 9 works only when the endpoint is in this format.



Settings

Device data endpoint

Your devices can use your account's device data endpoint to connect to AWS.

Each of your things has a REST API available at this endpoint. MQTT clients and AWS IoT Device SDKs also use this endpoint.

Endpoint: a2gbh2bus0sam1-ats.iot.us-east-1.amazonaws.com

Domain configurations

You can create domain configurations to simplify tasks such as migrating devices to AWS IoT Core, migrating application infrastructure to AWS IoT Core and maintaining brand identity.

Name	Domain name	Status	Service type	Date updated

No domain configurations

You don't have any domain configurations.

Logs

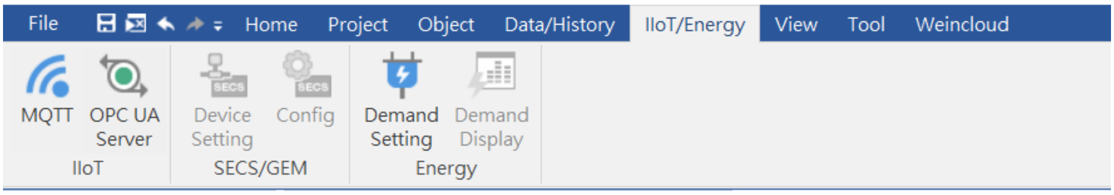
Looking for language selection? Find it in the new Unified Settings

© 2022, Amazon Web Services, Inc. or its affiliates.

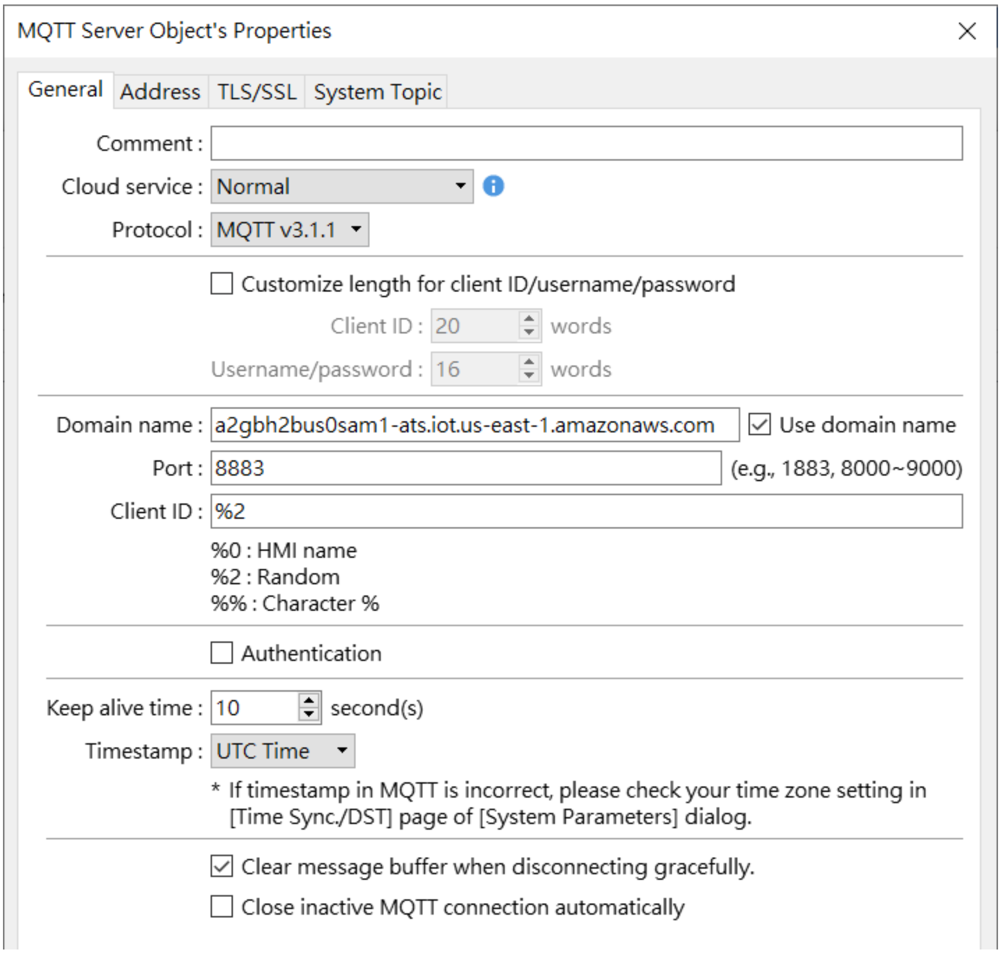
3. EasyBuilder Pro Settings

After hosting an MQTT server, launch EasyBuilder Pro.

1. Click [IIoT/Energy] » [MQTT] to open MQTT settings window.



2. In General tab, select [Normal] as cloud service to use publish-subscribe mode, or select [AWS IoT] to use Thing mode, and the rest will be introduced later. Use the URL obtained in Chapter 2 as domain name, and use port 8883.


MQTT Server Object's Properties

Comment :

Cloud service : Normal

Protocol : MQTT v3.1.1

□ Customize length for client ID/username/password

Client ID : 20 words

Username/password : 16 words

Domain name : a2gbh2bus0sam1-ats.iot.us-east-1.amazonaws.com ☑ Use domain name

Port : 8883 (e.g., 1883, 8000~9000)

Client ID : %2

%0 : HMI name
%2 : Random
%% : Character %

□ Authentication

Keep alive time : 10 second(s)

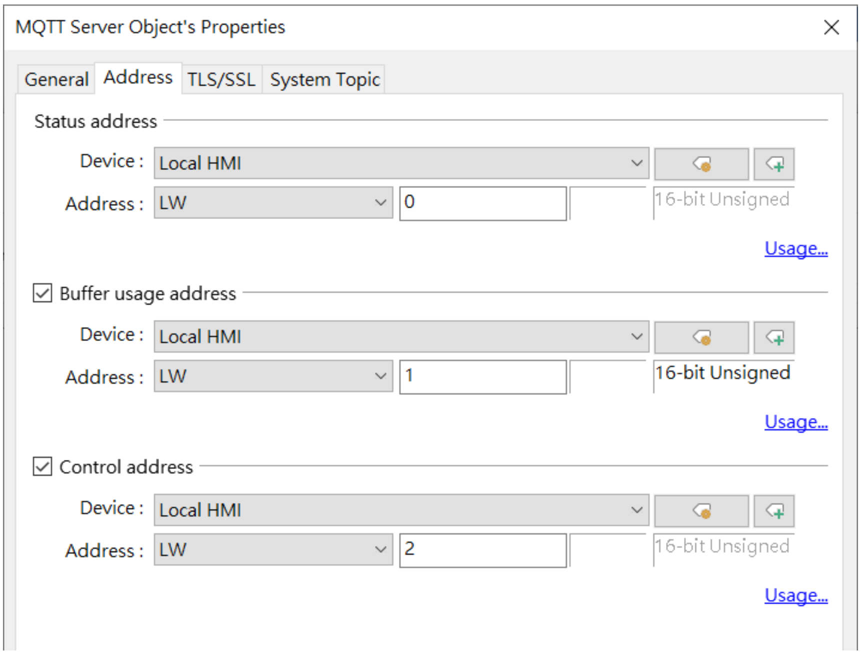
Timestamp : UTC Time

* If timestamp in MQTT is incorrect, please check your time zone setting in [Time Sync./DST] page of [System Parameters] dialog.

☑ Clear message buffer when disconnecting gracefully.

□ Close inactive MQTT connection automatically

3. In Address tab configure addresses.



MQTT Server Object's Properties

Status address

Device : Local HMI

Address : LW 0 16-bit Unsigned

☑ Buffer usage address

Device : Local HMI

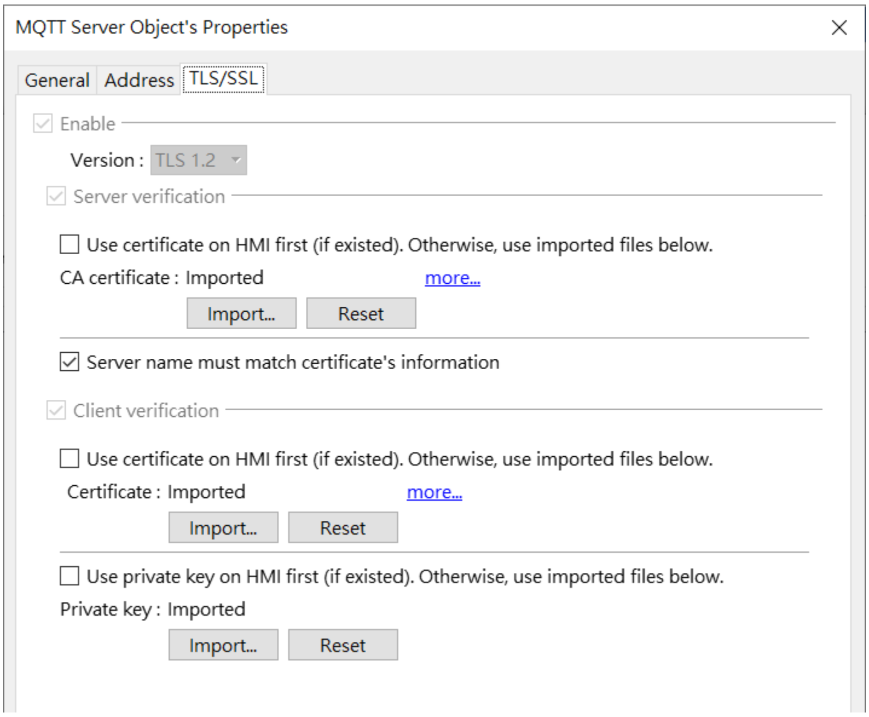
Address : LW 1 16-bit Unsigned

☑ Control address

Device : Local HMI

Address : LW 2 16-bit Unsigned

4. In TLS/SSL tab, import the file generated when creating the certificate.
Server verification, CA certificate: Import a .pem file. (Amazon Root CA 1)
Client verification, Certificate: Import a .crt file. (certificate.pem.crt)
Client verification, Private key: Import a .key file (private.pem.key)



MQTT Server Object's Properties

☑ Enable

Version : TLS 1.2

☑ Server verification

□ Use certificate on HMI first (if existed). Otherwise, use imported files below.

CA certificate : Imported

☑ Server name must match certificate's information

☑ Client verification

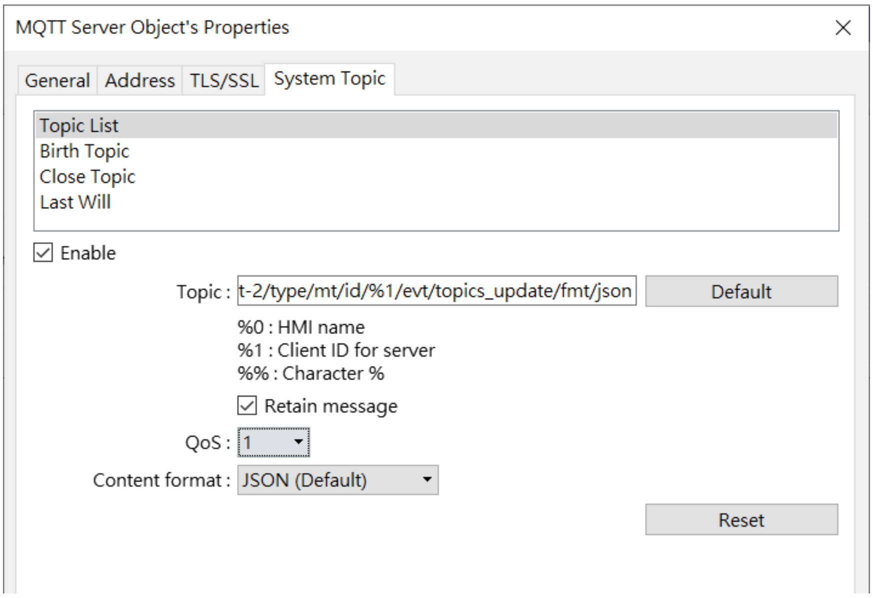
□ Use certificate on HMI first (if existed). Otherwise, use imported files below.

Certificate : Imported

□ Use private key on HMI first (if existed). Otherwise, use imported files below.

Private key : Imported

5. System topic includes Topic List and Connection State that HMI will automatically send once it connects to server.



MQTT Server Object's Properties

Topic List
Birth Topic
Close Topic
Last Will

☑ Enable

Topic : t-2/type/mt/id/%1/evt/topics_update/fmt/json

%0 : HMI name
%1 : Client ID for server
%% : Character %

☑ Retain message

QoS : 1

Content format : JSON (Default)

6. Restrictions of using AWS IoT as MQTT server:
   a. Only QoS 0 and QoS 1 are available.
   b. Retain message is not supported.
   c. The maximum number of layers is 8.
7. Please see EasyBuilder Pro user manual for more information about publish / subscribe settings.

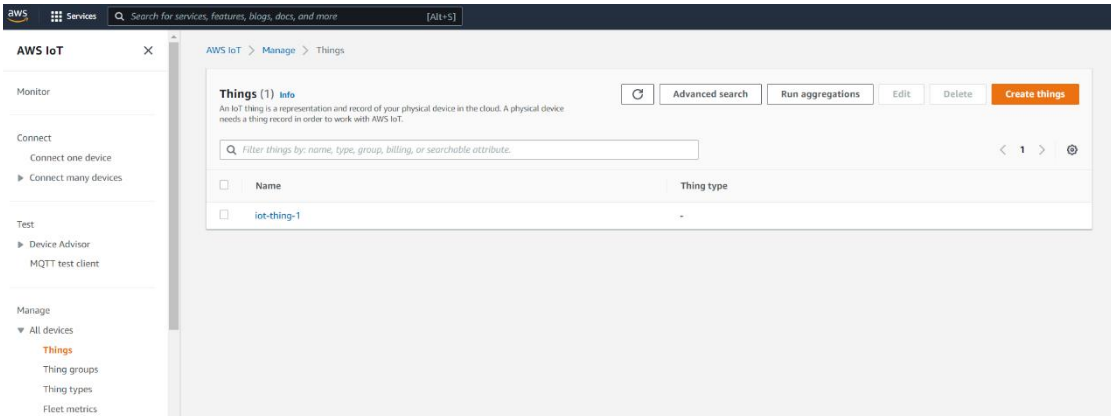
4. Thing and Shadow

With AWS IoT, Publisher->Broker-> Subscribe is no longer the only path that data is accessed over MQTT. By introducing Thing Shadow service, a Thing (a device, app…etc) can interact with cloud applications and other devices connected to AWS IoT. A Shadow can be maintained for each Thing connected to AWS IoT. The Shadow can be used to get/set the state of a Thing over MQTT, regardless of whether the Thing is connected to the Internet.

This chapter explains configuration of AWS IoT and EasyBuilder Pro.

AWS IoT

1. Click [Manage] » [All devices] » [Things] » [Create things].

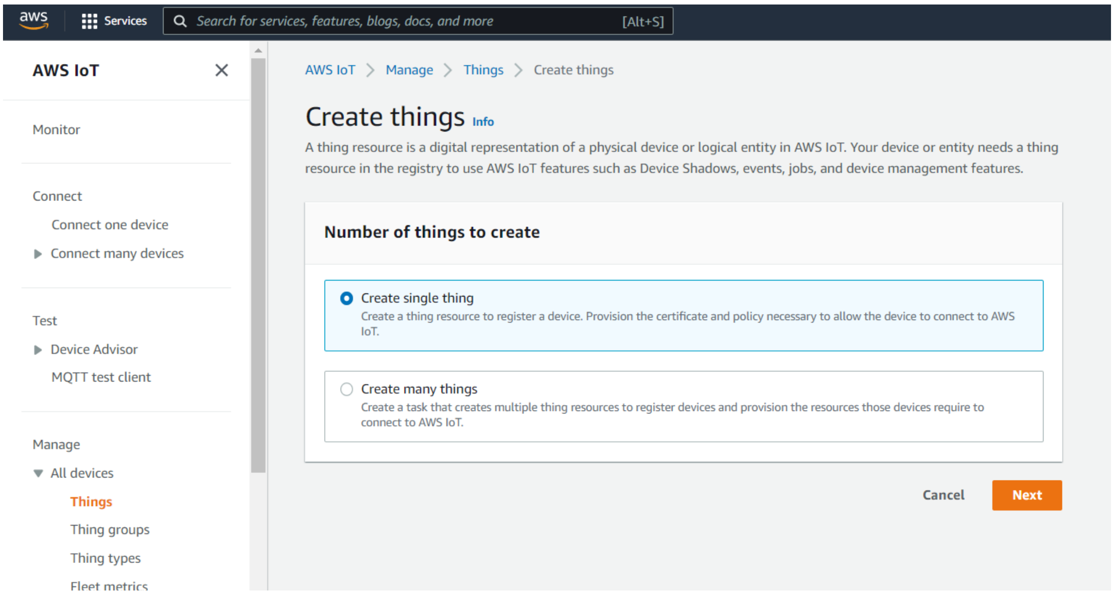


Things (1) Info

An IoT thing is a representation and record of your physical device in the cloud. A physical device needs a thing record in order to work with AWS IoT.

Name	Thing type
iot-thing-1	-

2. Select [Create single thing] and then click [Next].



Create things Info

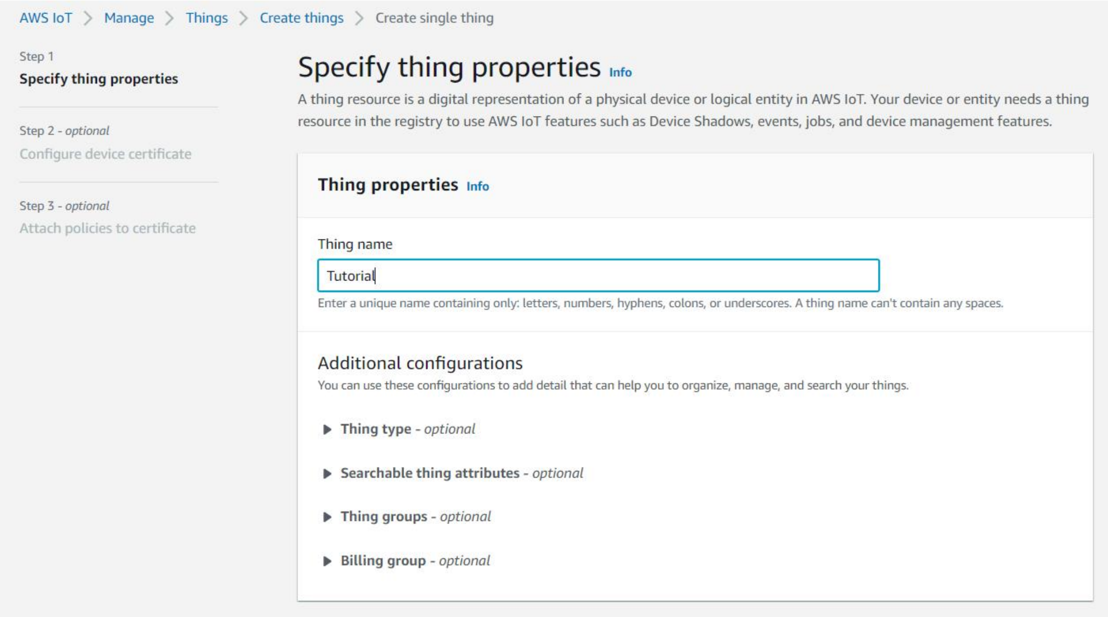
A thing resource is a digital representation of a physical device or logical entity in AWS IoT. Your device or entity needs a thing resource in the registry to use AWS IoT features such as Device Shadows, events, jobs, and device management features.

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.

3. Enter the Thing name.



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: Tutorial

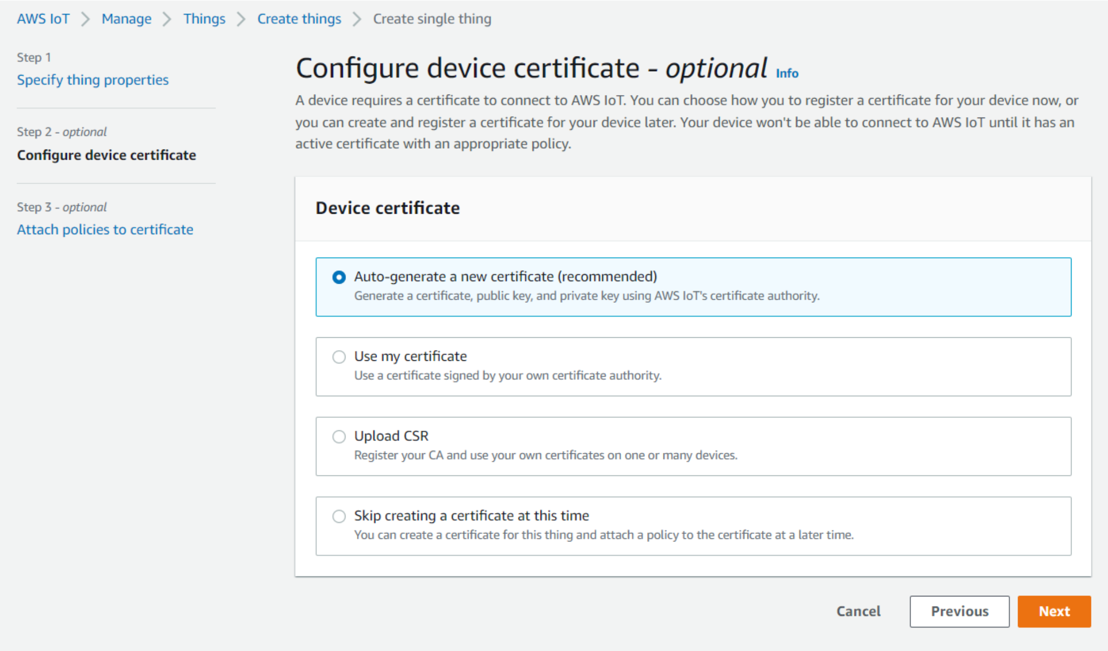
Enter a unique name containing only: letters, numbers, hyphens, colons, or underscores. A thing name can't contain any spaces.

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*

4. Create a certificate.



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

Configure device certificate - *optional* Info

A device requires a certificate to connect to AWS IoT. You can choose how you 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

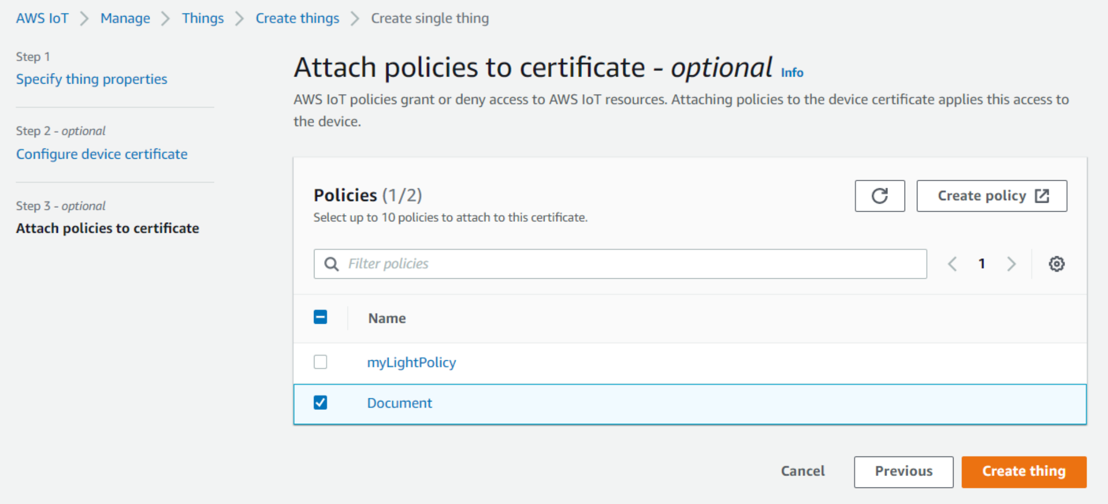
● Auto-generate a new certificate (recommended)
Generate a certificate, public key, and private key using AWS IoT's certificate authority.

○ 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.

5. Select the corresponding Policy and then click [Create thing].



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

Attach policies to certificate - *optional* Info

AWS IoT policies grant or deny access to AWS IoT resources. Attaching policies to the device certificate applies this access to the device.

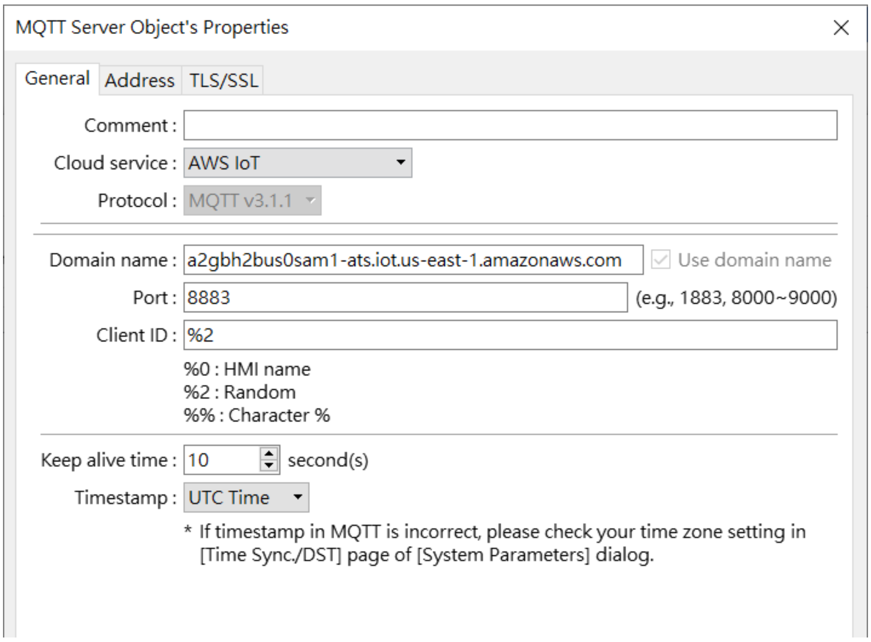
Policies (1/2)

Select up to 10 policies to attach to this certificate.

	Name
□	myLightPolicy
☑	Document

EasyBuilder Pro

1. Select AWS IoT as cloud service in MQTT Server settings window, and follow the settings in Chapter 3 in this manual.



MQTT Server Object's Properties

Comment :

Cloud service : AWS IoT

Protocol : MQTT v3.1.1

Domain name : a2gbh2bus0sam1-ats.iot.us-east-1.amazonaws.com ☑ Use domain name

Port : 8883 (e.g., 1883, 8000~9000)

Client ID : %2

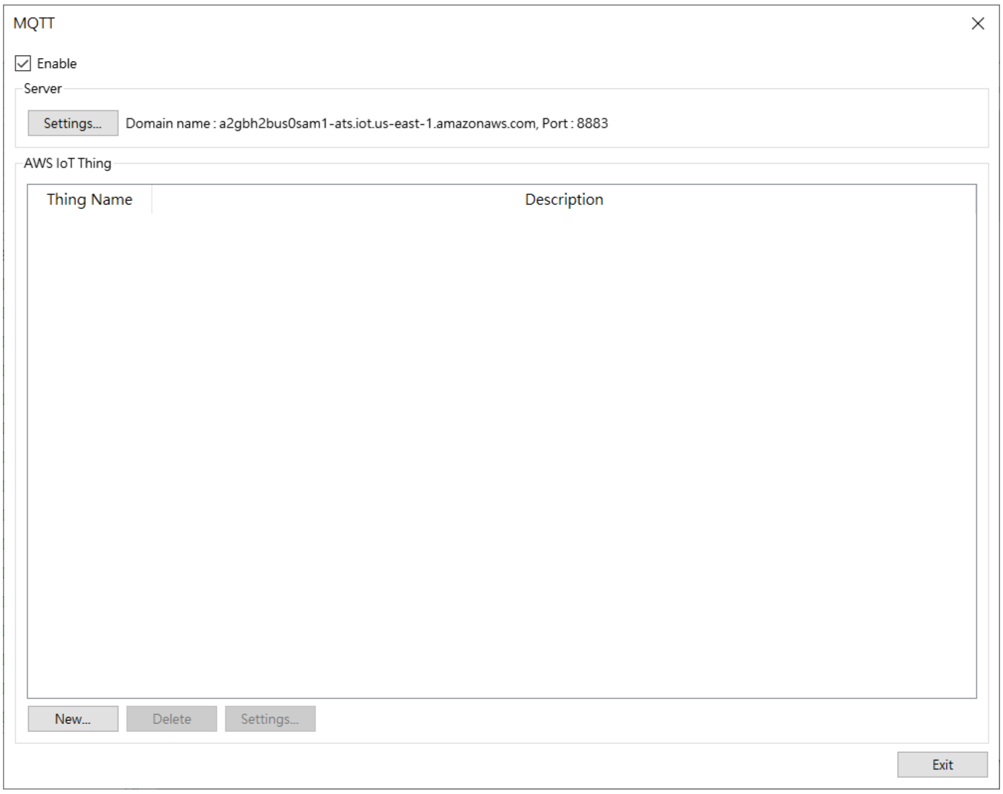
%0 : HMI name
%2 : Random
%% : Character %

Keep alive time : 10 second(s)

Timestamp : UTC Time

* If timestamp in MQTT is incorrect, please check your time zone setting in [Time Sync./DST] page of [System Parameters] dialog.

2. Click new to add a device.



MQTT

☑ Enable

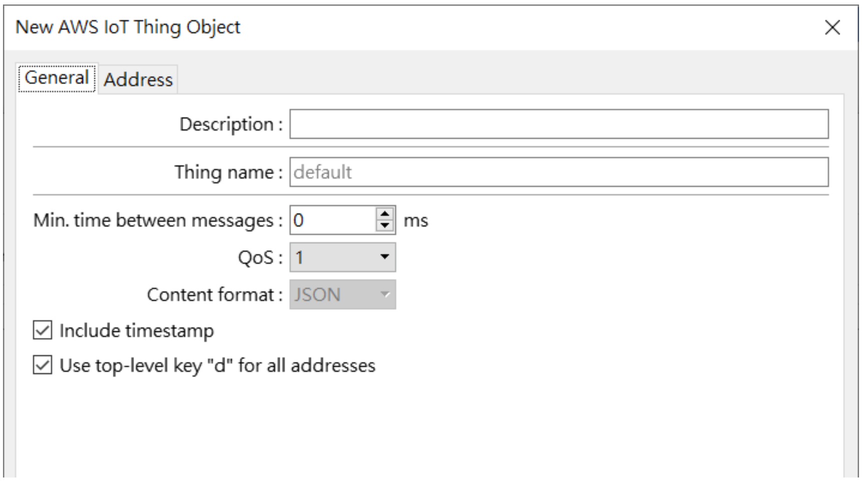
Server

Settings... Domain name : a2gbh2bus0sam1-ats.iot.us-east-1.amazonaws.com, Port : 8883

AWS IoT Thing

Thing Name	Description

3. Enter Thing name and set minimal time between messages. Only Qos 0 and Qos 1 are available.



New AWS IoT Thing Object

Description :

Thing name : default

Min. time between messages : 0 ms

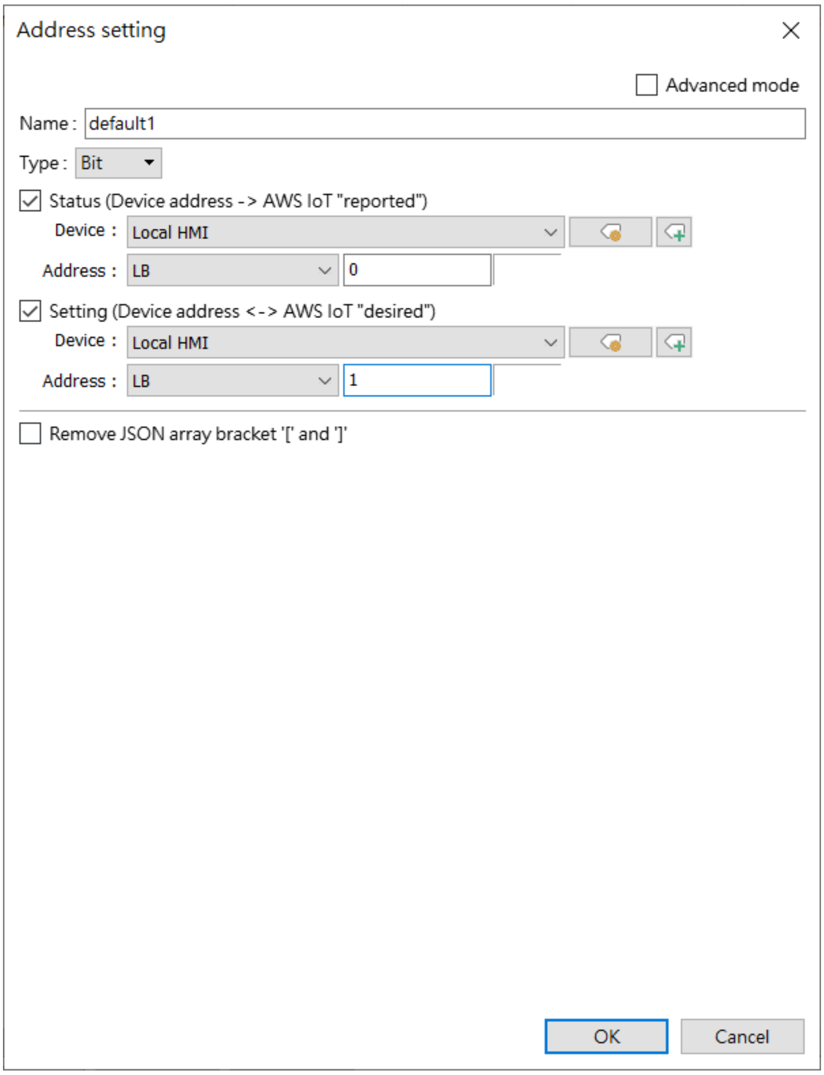
QoS : 1

Content format : JSON

☑ Include timestamp

☑ Use top-level key "d" for all addresses

4. Go to Address tab and set the addresses for reported status (LB-0) and desired setting (LB-1). ->, <-> stands for the direction in which data is transmitted.



Address setting

□ Advanced mode

Name : default1

Type : Bit

☑ Status (Device address -> AWS IoT "reported")

Device : Local HMI

Address : LB 0

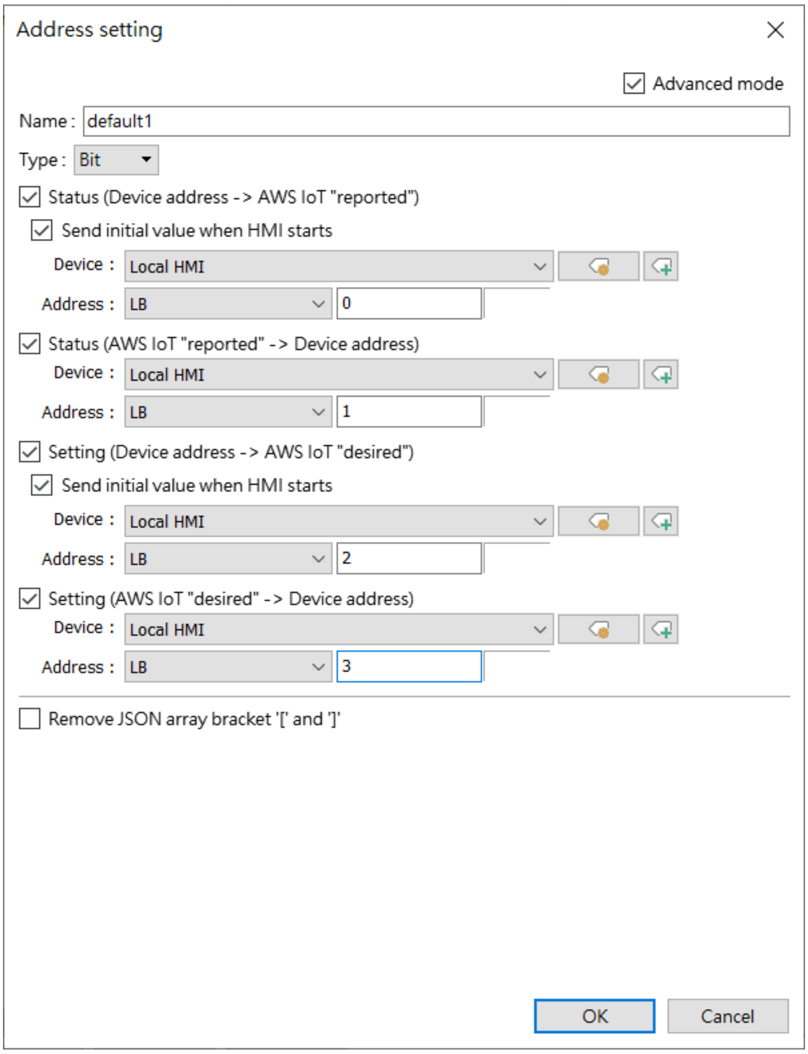
☑ Setting (Device address <-> AWS IoT "desired")

Device : Local HMI

Address : LB 1

□ Remove JSON array bracket '[' and ']'

5. In Advanced Mode settings window, Status (reported) and Setting (desired) can use different addresses, and data is transmitted to/from AWS IoT/device.



Address setting

☑ Advanced mode

Name : default1

Type : Bit

☑ Status (Device address -> AWS IoT "reported")

☑ Send initial value when HMI starts

Device : Local HMI

Address : LB 0

☑ Status (AWS IoT "reported" -> Device address)

Device : Local HMI

Address : LB 1

☑ Setting (Device address -> AWS IoT "desired")

☑ Send initial value when HMI starts

Device : Local HMI

Address : LB 2

☑ Setting (AWS IoT "desired" -> Device address)

Device : Local HMI

Address : LB 3

□ Remove JSON array bracket '[' and ']'

5. References

- EasyBuilder Pro User Manual Chapter 42 IIoT
- How does AWS IoT platform work: https://aws.amazon.com/tw/iot-platform/how-it-works/