This document describes how to configure AWS IoT Core and AWS IoT Greengrass to run edge inference application on LG AIoT Board.

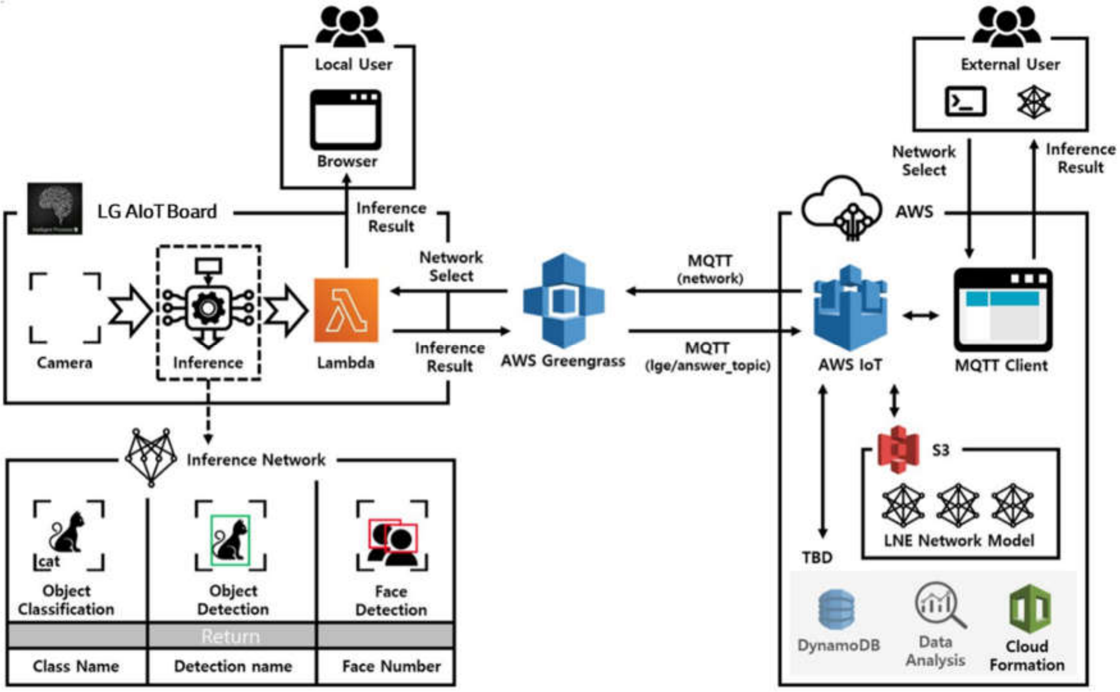


Figure 1. Demo application architecture of AWS Greengrass with LG AIoT Board

[Step1] Setting up a LG AIoT Board

Follow "LG8111 uses AWS Greengrass on its development board" section of the "AWS Getting Started with LG AIoT Board to use AWS IoT Greengrass on LG AIoT Board

[Step2] Run the edge inferencing application on the LG AIoT Board

1. How to set the downloaded core resource

Transfer the Core Resource (*Hash*-setup.tar.gz, e.g.:e892362d7d-setup.tar.gz) from your computer to the LG AIoT Board. Open a terminal window on your computer and run the following commands.

```
scp Hash-setup.tar.gz ubuntu@IP-address:/home/ubuntu
```

Open a terminal on the LG AIoT Board and navigate to the folder that contains the compressed files. (cd /home/ubuntu) Decompress the Core Resource by entering the following command.

```
sudo tar -xzvf Hash-setup.tar.gz -C /greengrass
```

Download the Root CA certificate to LG AIoT Board's /greengrass/certs folder to communicate with AWS IoT Core.

```
cd /greengrass/certs/
sudo wget -O root.ca.pem https://www.amazontrust.com/repository/AmazonRootCA1.pem
```

2. Lambda Code generation

You need to download Lambda code from https://github.com/lge-aws-dist/aws_gg_lambda and compress files using following command.

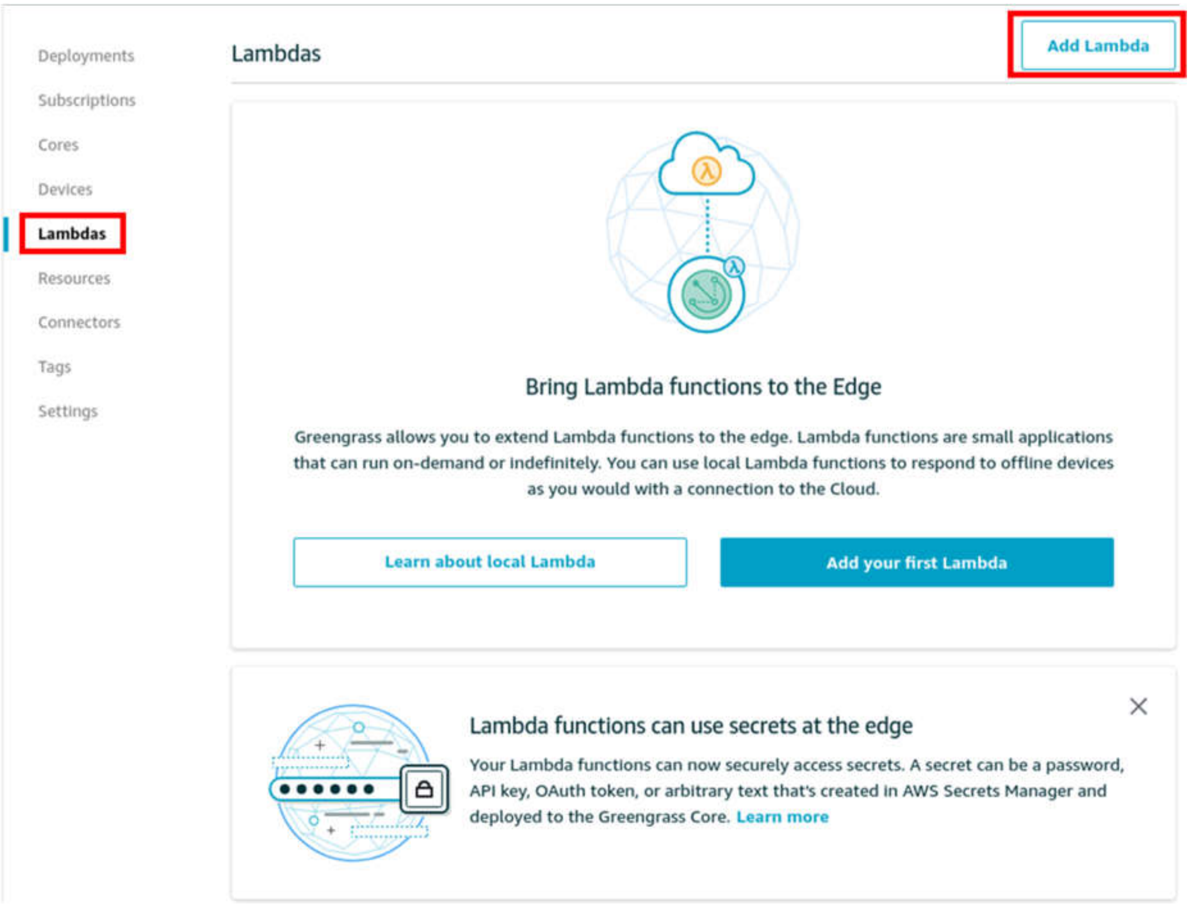
```
$ zip -r greengrassML.zip greengrasssdk templates labels lib network greengrassML.py
```

3. Upload Network model acceptable by the LNE(.lne) to S3

The application is configured to work with three ML network models.(Tiny-yolo, MobileNet, and Mtcnn) These network models should be a format allowed by LNE, and need to be uploaded into AWS S3. When setting machine resource of AWS IoT Greengrass, you can set it to refer to the model uploaded to S3.

4. Lambda Registration

On AWS IoT Greengress console, select Lambda -> Add Lambda to register a new Lambda.



Lambdas

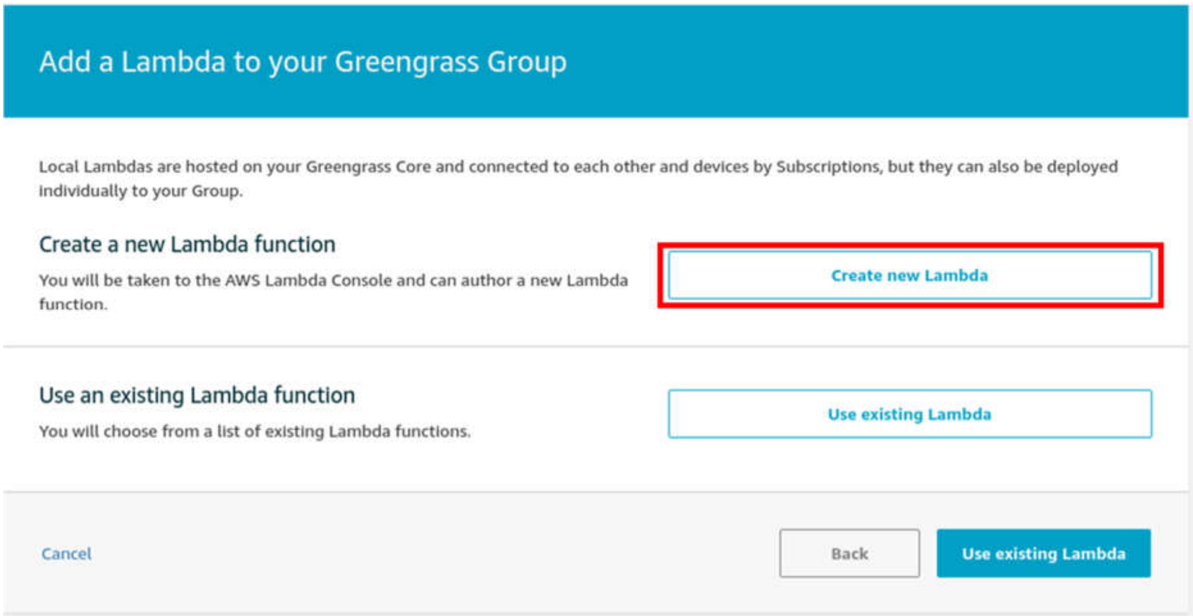
Bring Lambda functions to the Edge

Greengrass allows you to extend Lambda functions to the edge. Lambda functions are small applications that can run on-demand or indefinitely. You can use local Lambda functions to respond to offline devices as you would with a connection to the Cloud.

Lambda functions can use secrets at the edge

Your Lambda functions can now securely access secrets. A secret can be a password, API key, OAuth token, or arbitrary text that's created in AWS Secrets Manager and deployed to the Greengrass Core. Learn more

Select [Create new Lambda] to add a Lambda function to AWS IoT configuration.



Add a Lambda to your Greengrass Group

Local Lambdas are hosted on your Greengrass Core and connected to each other and devices by Subscriptions, but they can also be deployed individually to your Group.

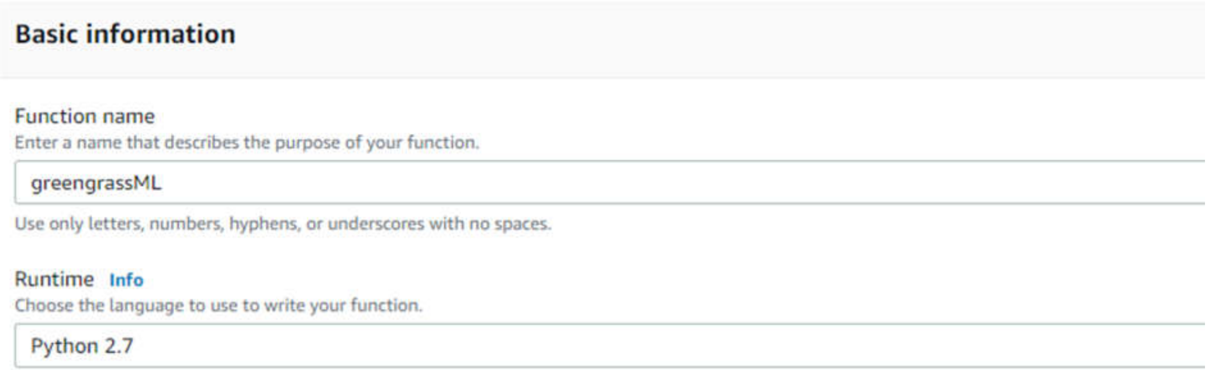
Create a new Lambda function

You will be taken to the AWS Lambda Console and can author a new Lambda function.

Use an existing Lambda function

You will choose from a list of existing Lambda functions.

The Function name is an example of a Python 2.7, registered with a Lambda-distinctive name and created based on python 2.7, so select the Runtime option to python 2.7, then move on to the next stage by clicking Create Function.



Basic information

Function name

Enter a name that describes the purpose of your function.

greengrassML

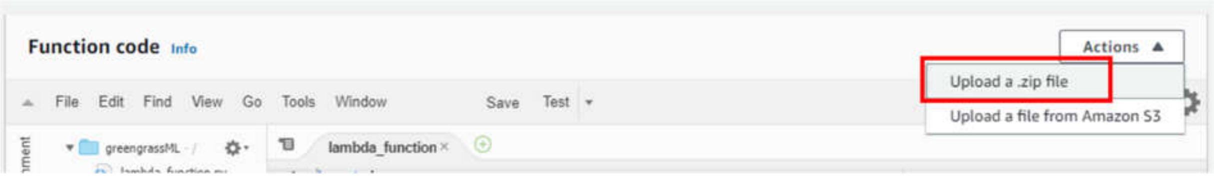
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime Info

Choose the language to use to write your function.

Python 2.7

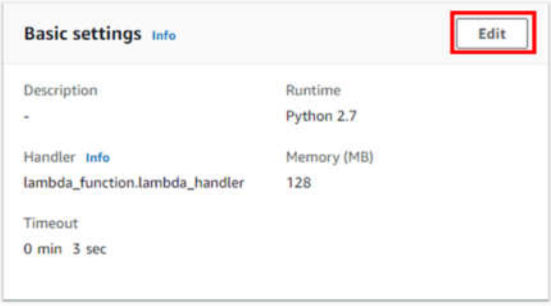
The Lambda function that runs AWS IoT Greengrass on LG AIoT Board is in .zip format, so code entry type is zip. Upload the generated Lambda code by selecting “Upload a .zip file”.



Function code Info

The Lambda function runtime and Handler need to be set like below.

- Runtime : python 2.7
- Handler : "[Lambda file name].[Handler name in Lambda file]”



Basic settings Info

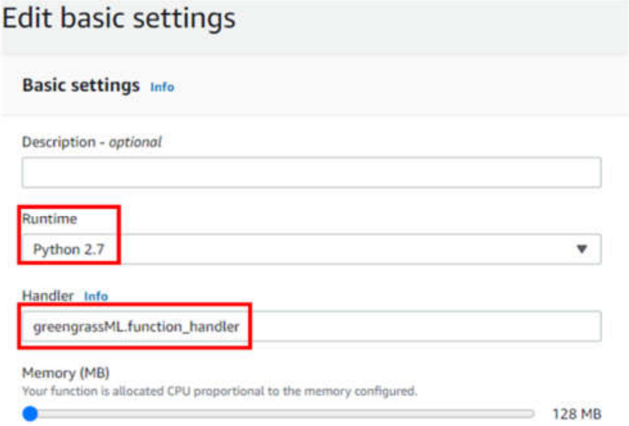
Description
-

Runtime
Python 2.7

Handler Info
lambda_function.lambda_handler

Memory (MB)
128

Timeout
0 min 3 sec



Edit basic settings

Basic settings Info

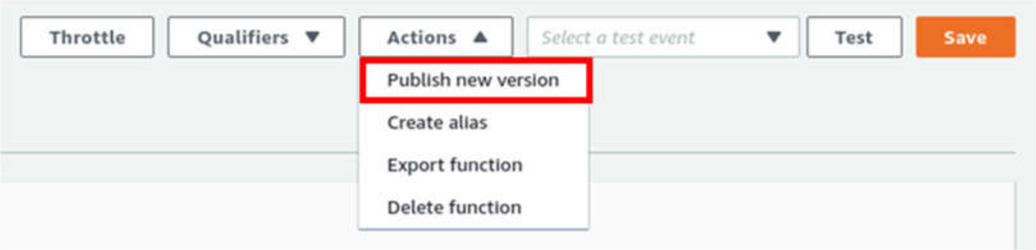
Description - *optional*

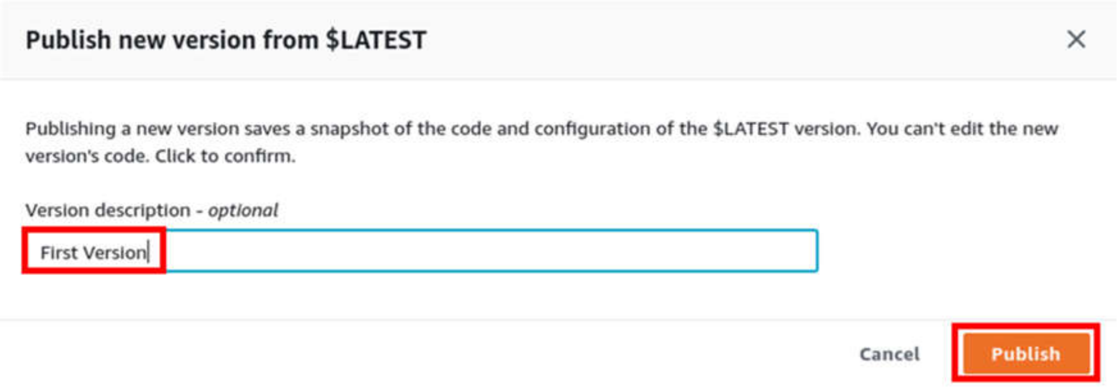
Runtime
Python 2.7

Handler Info
greengrassML.function_handler

Memory (MB)
Your function is allocated CPU proportional to the memory configured.
128 MB

Select Publish new version in Actions, and enter a description for the saved version.



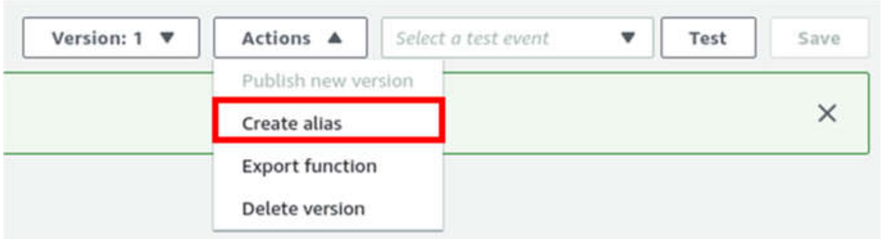

Publish new version from $LATEST

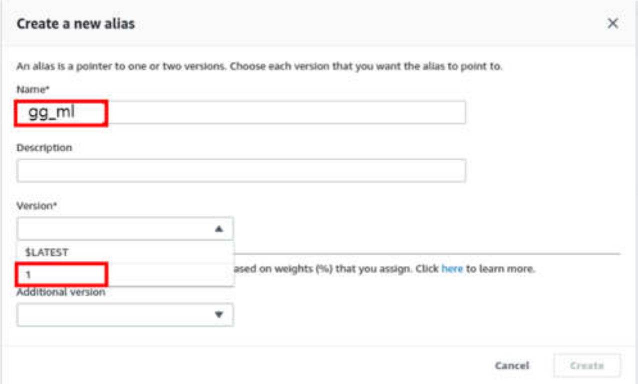
Publishing a new version saves a snapshot of the code and configuration of the $LATEST version. You can't edit the new version's code. Click to confirm.

Version description - *optional*

First Version

Saves the description for the version and saves it with alias and version for that version. (If you select Version with #LATEST, it may not work.)




Create a new alias

An alias is a pointer to one or two versions. Choose each version that you want the alias to point to.

Name*

gg_ml

Description

Version*

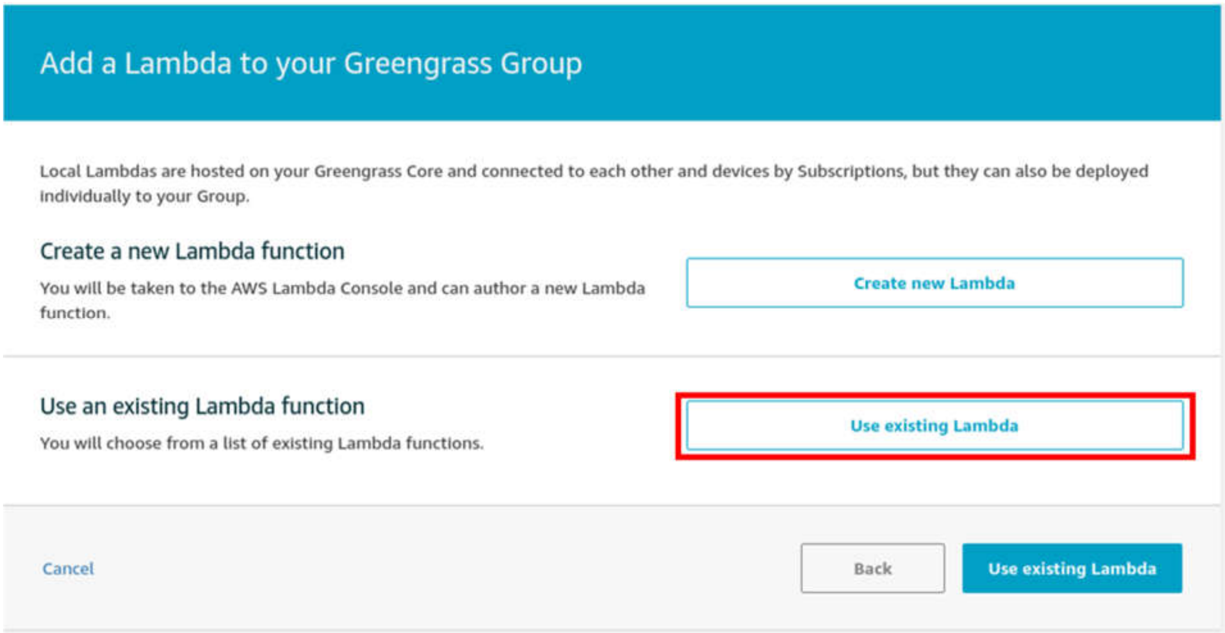
$LATEST

1

...ased on weights (%) that you assign. Click here to learn more.

Additional version

On AWS IoT Greengrass console, select Lambda->Add Lambda and select "Use existing Lambda" to use the saved lambda.



Add a Lambda to your Greengrass Group

Local Lambdas are hosted on your Greengrass Core and connected to each other and devices by Subscriptions, but they can also be deployed individually to your Group.

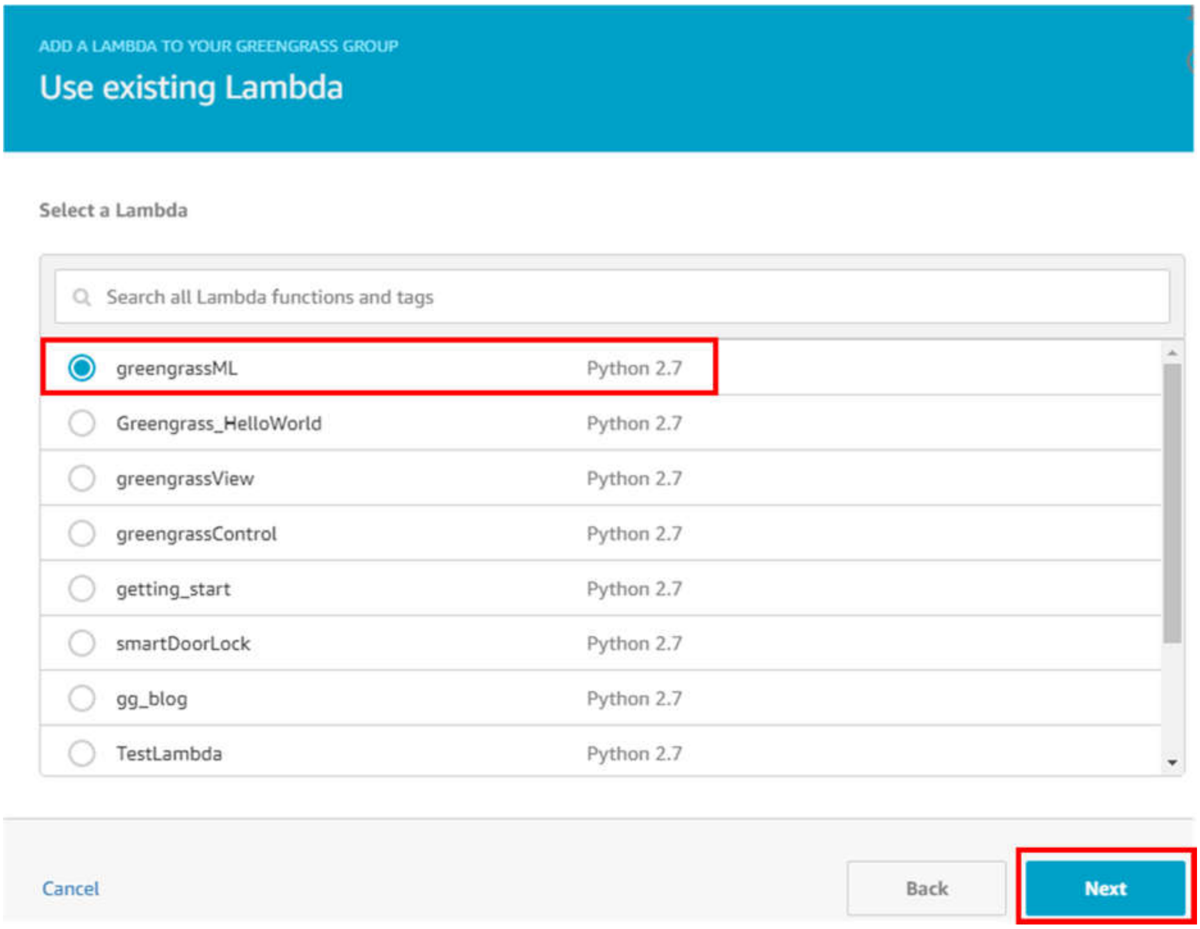
Create a new Lambda function

You will be taken to the AWS Lambda Console and can author a new Lambda function.

Use an existing Lambda function

You will choose from a list of existing Lambda functions.

Select Lambda name, Lambda version of Lambda that you registered for testing.



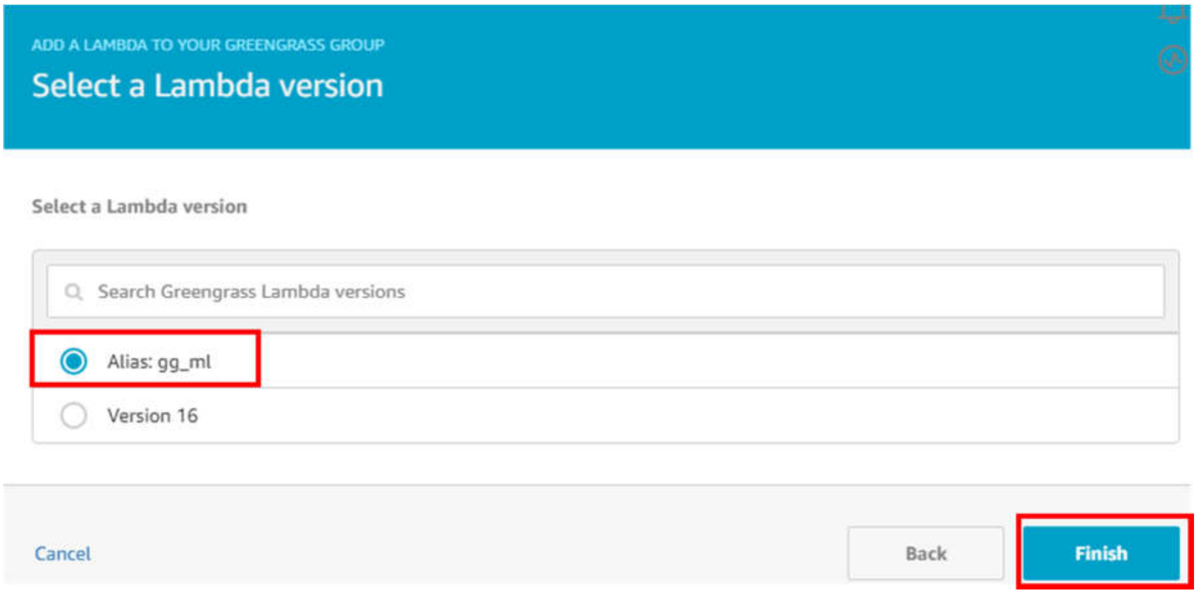
ADD A LAMBDA TO YOUR GREENGRASS GROUP

Use existing Lambda

Select a Lambda

Search all Lambda functions and tags

●	greengrassML	Python 2.7
○	Greengrass_HelloWorld	Python 2.7
○	greengrassView	Python 2.7
○	greengrassControl	Python 2.7
○	getting_start	Python 2.7
○	smartDoorLock	Python 2.7
○	gg_blog	Python 2.7
○	TestLambda	Python 2.7



ADD A LAMBDA TO YOUR GREENGRASS GROUP

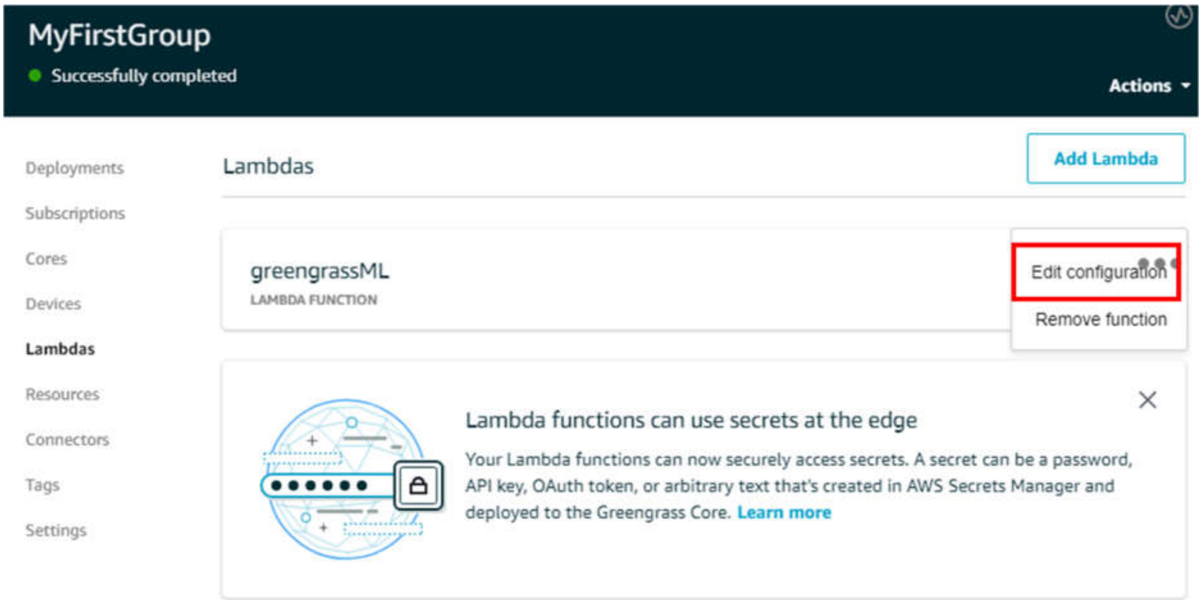
Select a Lambda version

Select a Lambda version

● Alias: gg_ml

○ Version 16

For the test, you must modify the configuration of the Lambda function that you registered. Select Edit configuration by pressing the three circles to the right of the Lambda function added for this purpose.



MyFirstGroup

● Successfully completed

Lambdas

greengrassML

LAMBDA FUNCTION

Lambda functions can use secrets at the edge

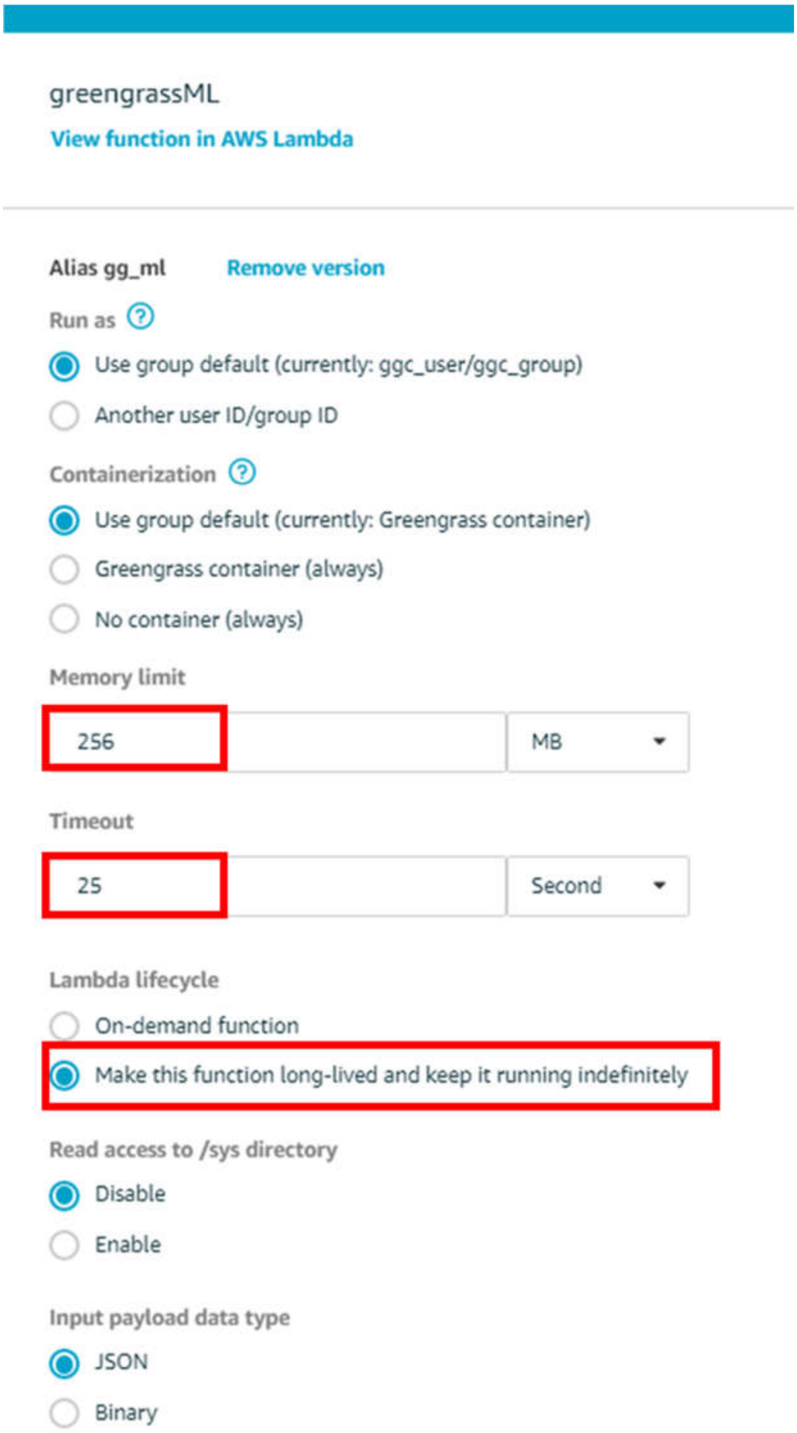
Your Lambda functions can now securely access secrets. A secret can be a password, API key, OAuth token, or arbitrary text that's created in AWS Secrets Manager and deployed to the Greengrass Core. Learn more

Correct the Memory limit, Timeout, and Lambda lifetime during the Lambda configuration.

Memory limit sets the memory limit to 256 MB when running the Lambda function.

Timeout is the Lambda function operation wait time, set to 25 seconds.

Lambda lifecycle saves the changes made by setting it to “Make this function log-live and keep it running indefinitely” running in the lifecycle setting of the Lambda function so that it can always operate for Test.



greengrassML

View function in AWS Lambda

Alias gg_ml Remove version

Run as

● Use group default (currently: ggc_user/ggc_group)

○ Another user ID/group ID

Containerization

● Use group default (currently: Greengrass container)

○ Greengrass container (always)

○ No container (always)

Memory limit

256 MB

Timeout

25 Second

Lambda lifecycle

○ On-demand function

● Make this function long-lived and keep it running indefinitely

Read access to /sys directory

● Disable

○ Enable

Input payload data type

● JSON

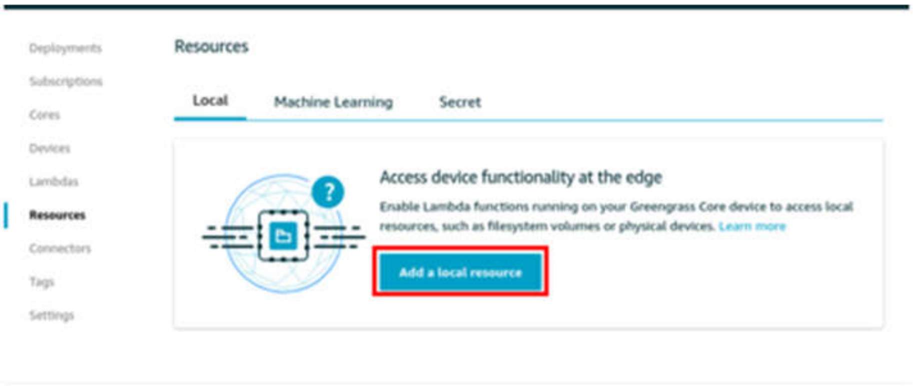
○ Binary

5. Local Resource Registration

Register the local resources (ex. camera, sensor, LNE, etc.) of the LG AIoT Board, which is used to run the AWS IoT Greengrass. The registered local resource is the local resource used by the Lambda function to operate.

NAME	DEVICE PATH	DESCRIPTION
LNE	/dev/dq1_lne	LG Neural HW Engine that efficiently processes deep learning algorithms (power saving, low latency)
CAMERA	/dev/video0	Camera on LG AIoT Board

Select the Local tab for the Resource to add local resource.

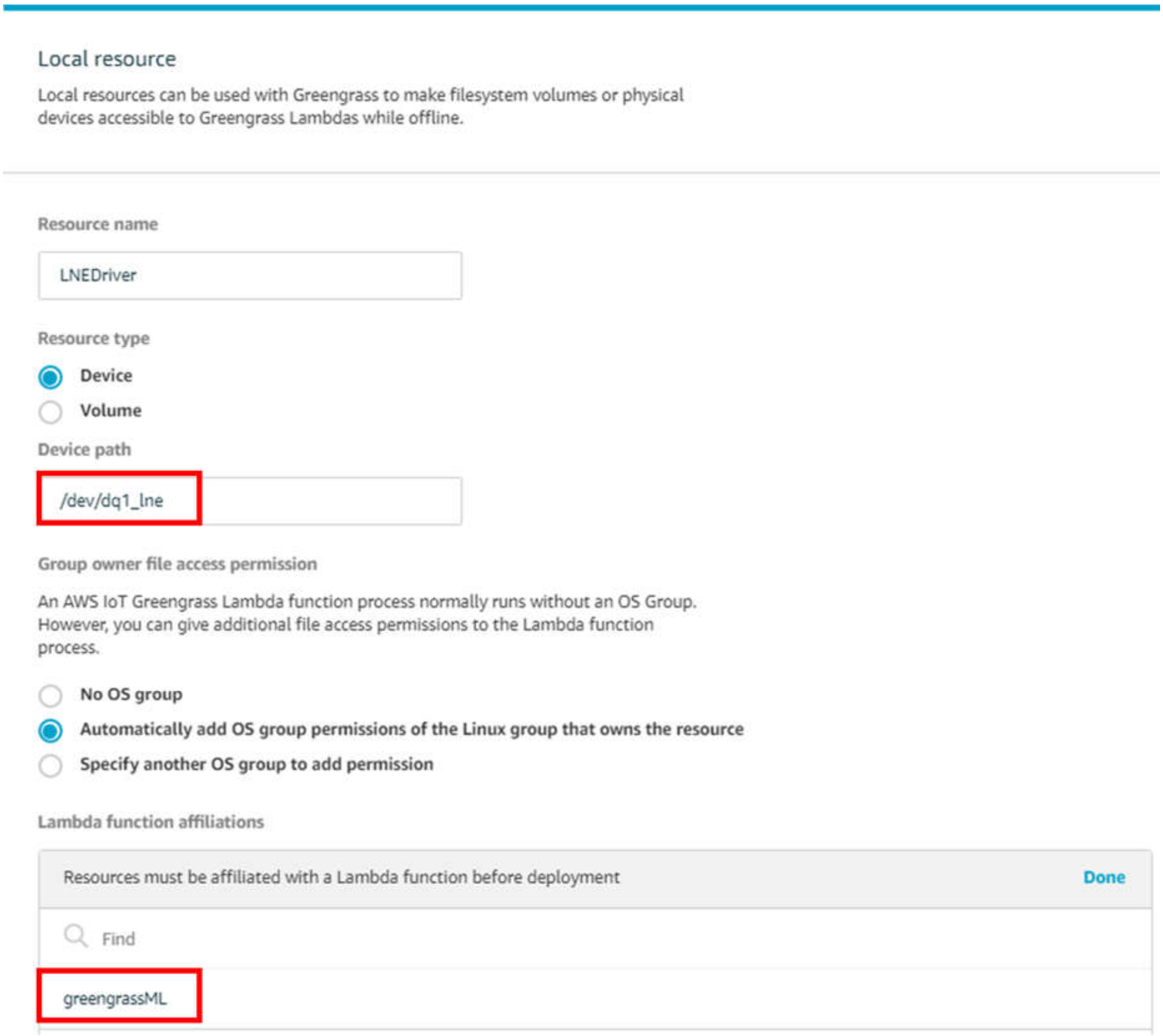


Access device functionality at the edge

Enable Lambda functions running on your Greengrass Core device to access local resources, such as filesystem volumes or physical devices. Learn more

4-1. Added LNE to Local Resource

Configure the device path and lambda function to use the local resource in order to add local resource for the LNE. At this time, set the setting for Lambda function to read and write access



Local resource

Local resources can be used with Greengrass to make filesystem volumes or physical devices accessible to Greengrass Lambdas while offline.

Resource name

LNEDriver

Resource type

● Device

○ Volume

Device path

/dev/dq1_lne

Group owner file access permission

An AWS IoT Greengrass Lambda function process normally runs without an OS Group. However, you can give additional file access permissions to the Lambda function process.

○ No OS group

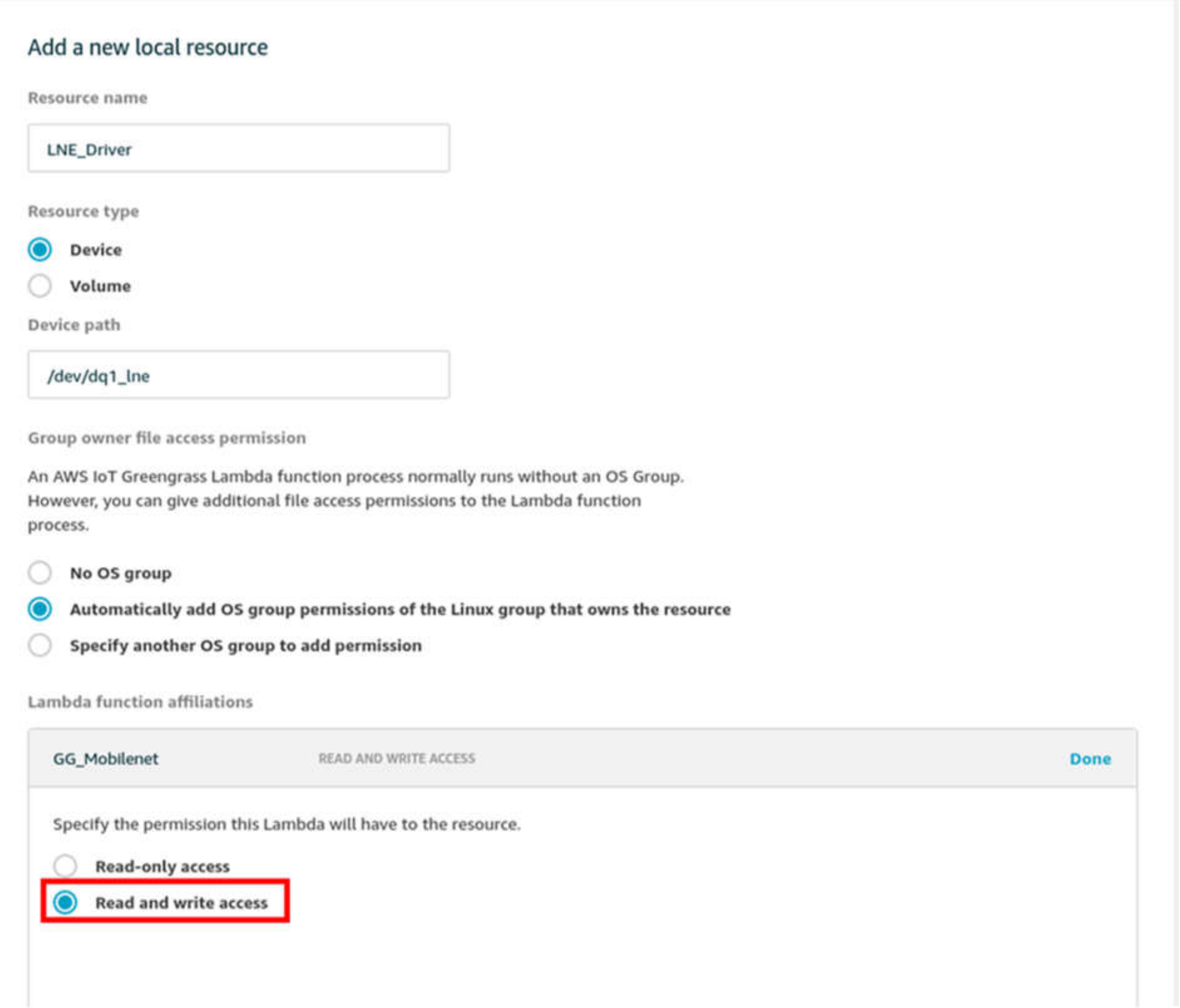
● Automatically add OS group permissions of the Linux group that owns the resource

○ Specify another OS group to add permission

Lambda function affiliations

Resources must be affiliated with a Lambda function before deployment

greengrassML



Add a new local resource

Resource name: LNE_Driver

Resource type
● Device
○ Volume

Device path: /dev/dq1_lne

Group owner file access permission

An AWS IoT Greengrass Lambda function process normally runs without an OS Group. However, you can give additional file access permissions to the Lambda function process.

○ No OS group
● Automatically add OS group permissions of the Linux group that owns the resource
○ Specify another OS group to add permission

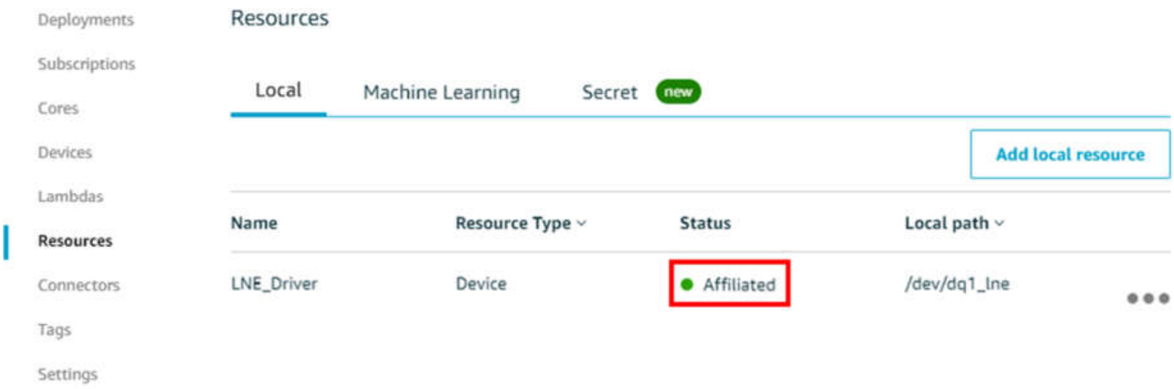
Lambda function affiliations

GG_Mobilenet — READ AND WRITE ACCESS — Done

Specify the permission this Lambda will have to the resource.

○ Read-only access
● Read and write access

After the local resource registration of the LG AI processor is completed, the status changes to green when it is successfully connected to Lambda function.

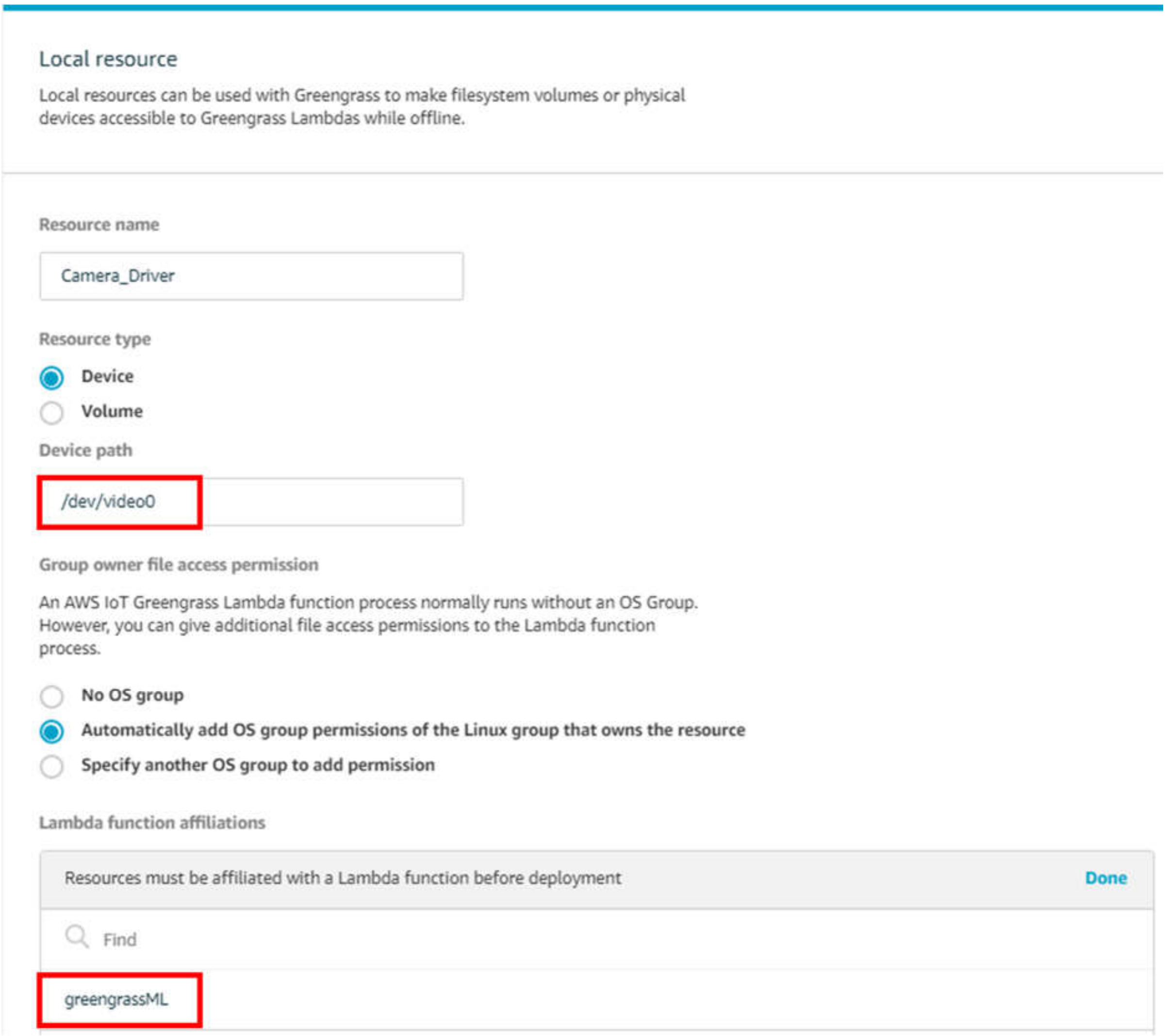


Resources

Name	Resource Type	Status	Local path
LNE_Driver	Device	● Affiliated	/dev/dq1_lne

4-2. Added Camera Local Resource

In order to use the LG8111 development board's Camera, the same process will be used to change only the Device path and register the LG AI processor as local resource.



Local resource

Local resources can be used with Greengrass to make filesystem volumes or physical devices accessible to Greengrass Lambdas while offline.

Resource name

Camera_Driver

Resource type

● Device

○ Volume

Device path

/dev/video0

Group owner file access permission

An AWS IoT Greengrass Lambda function process normally runs without an OS Group. However, you can give additional file access permissions to the Lambda function process.

○ No OS group

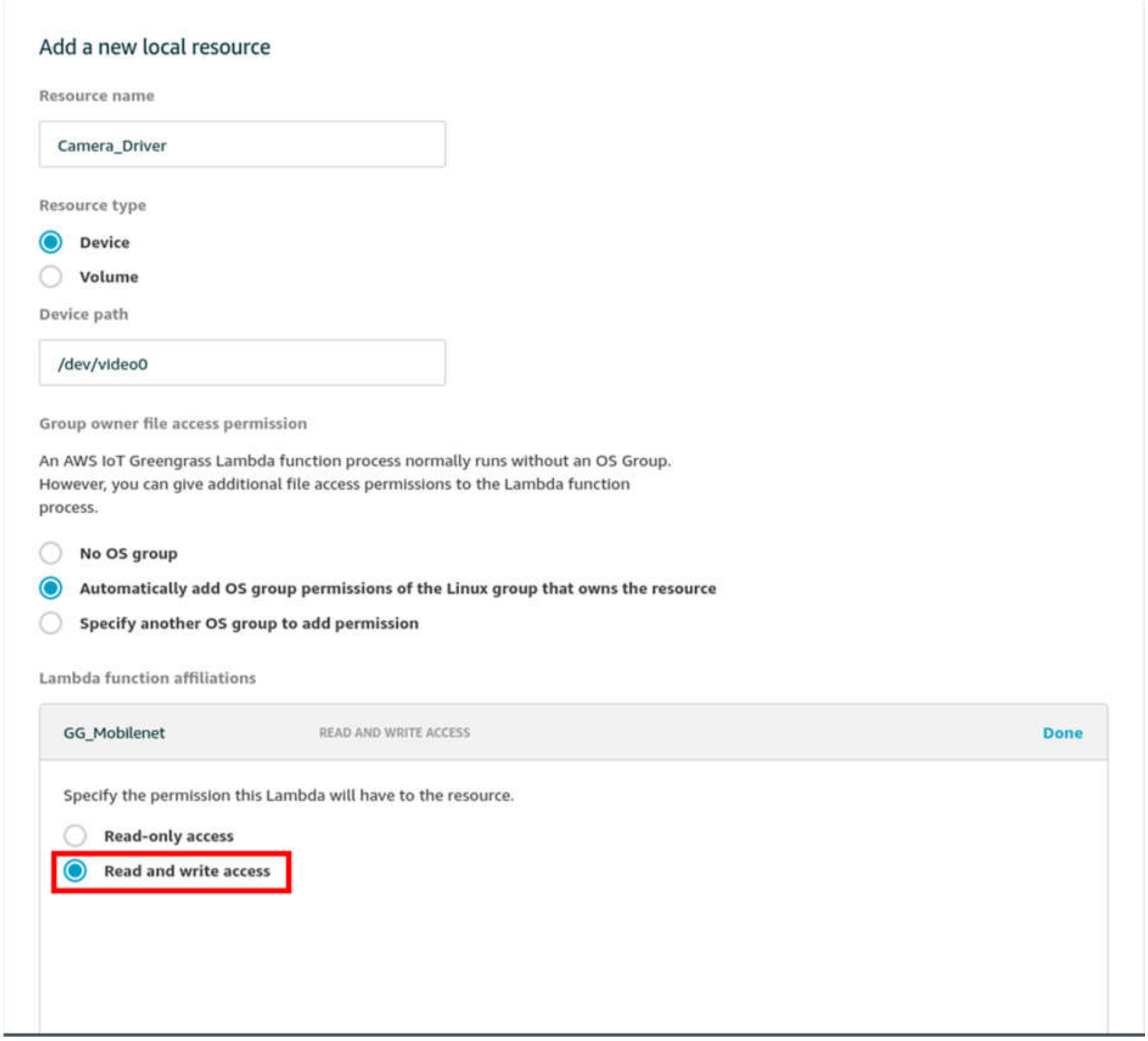
● Automatically add OS group permissions of the Linux group that owns the resource

○ Specify another OS group to add permission

Lambda function affiliations

Resources must be affiliated with a Lambda function before deployment

greengrassML



Add a new local resource

Resource name

Camera_Driver

Resource type

● Device

○ Volume

Device path

/dev/video0

Group owner file access permission

An AWS IoT Greengrass Lambda function process normally runs without an OS Group. However, you can give additional file access permissions to the Lambda function process.

○ No OS group

● Automatically add OS group permissions of the Linux group that owns the resource

○ Specify another OS group to add permission

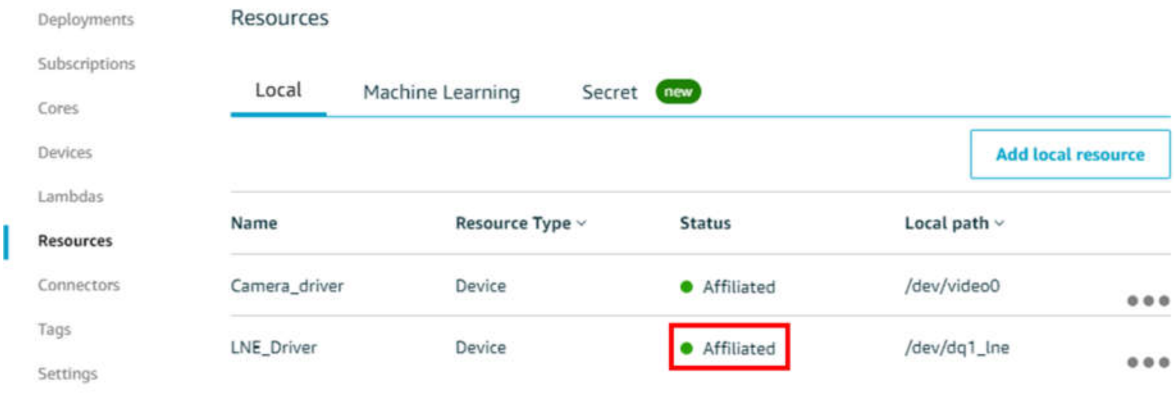
Lambda function affiliations

GG_Mobilenet READ AND WRITE ACCESS

Specify the permission this Lambda will have to the resource.

○ Read-only access

● Read and write access

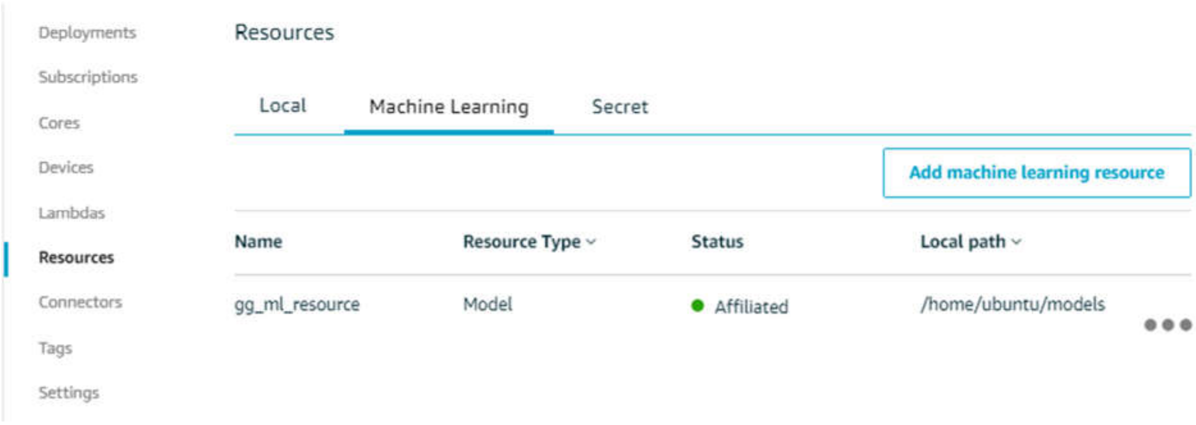


Resources

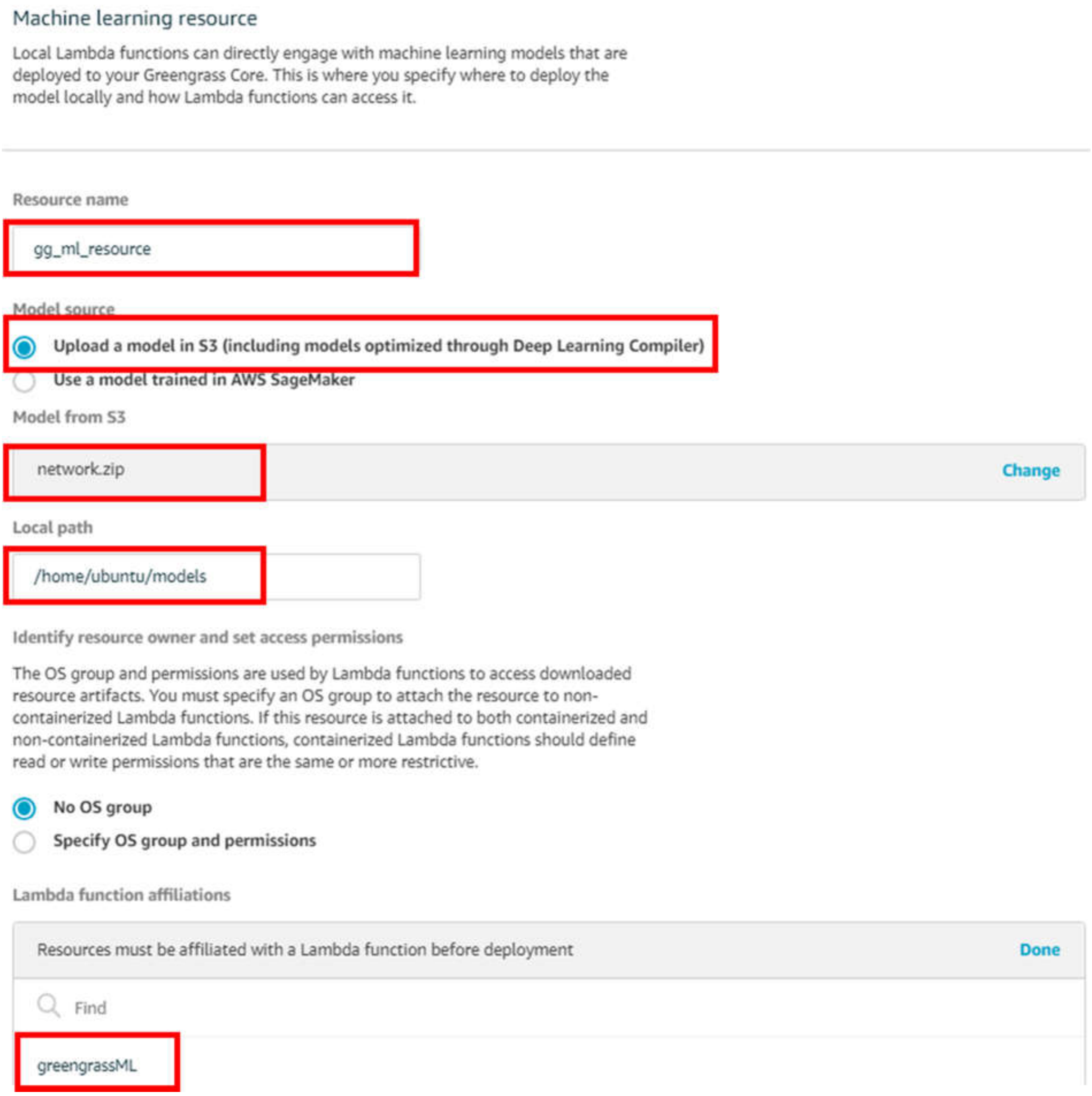
Name	Resource Type	Status	Local path
Camera_driver	Device	● Affiliated	/dev/video0
LNE_Driver	Device	● Affiliated	/dev/dq1_lne

6. Add Machine learning resource

The application requires network models uploaded in AWS S3 to perform local inferencing. By configuring the machine learning resource on AWS IoT Greengrass, the network models are can be deployed on LG AIoT Board.



Name	Resource Type	Status	Local path
gg_ml_resource	Model	● Affiliated	/home/ubuntu/models



Machine learning resource

Local Lambda functions can directly engage with machine learning models that are deployed to your Greengrass Core. This is where you specify where to deploy the model locally and how Lambda functions can access it.

Resource name

gg_ml_resource

Model source

● Upload a model in S3 (including models optimized through Deep Learning Compiler)

○ Use a model trained in AWS SageMaker

Model from S3

network.zip

Local path

/home/ubuntu/models

Identify resource owner and set access permissions

The OS group and permissions are used by Lambda functions to access downloaded resource artifacts. You must specify an OS group to attach the resource to non-containerized Lambda functions. If this resource is attached to both containerized and non-containerized Lambda functions, containerized Lambda functions should define read or write permissions that are the same or more restrictive.

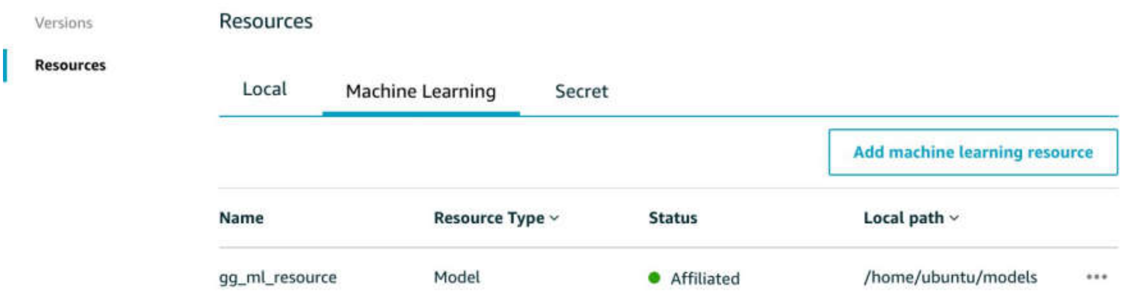
● No OS group

○ Specify OS group and permissions

Lambda function affiliations

Resources must be affiliated with a Lambda function before deployment

greengrassML

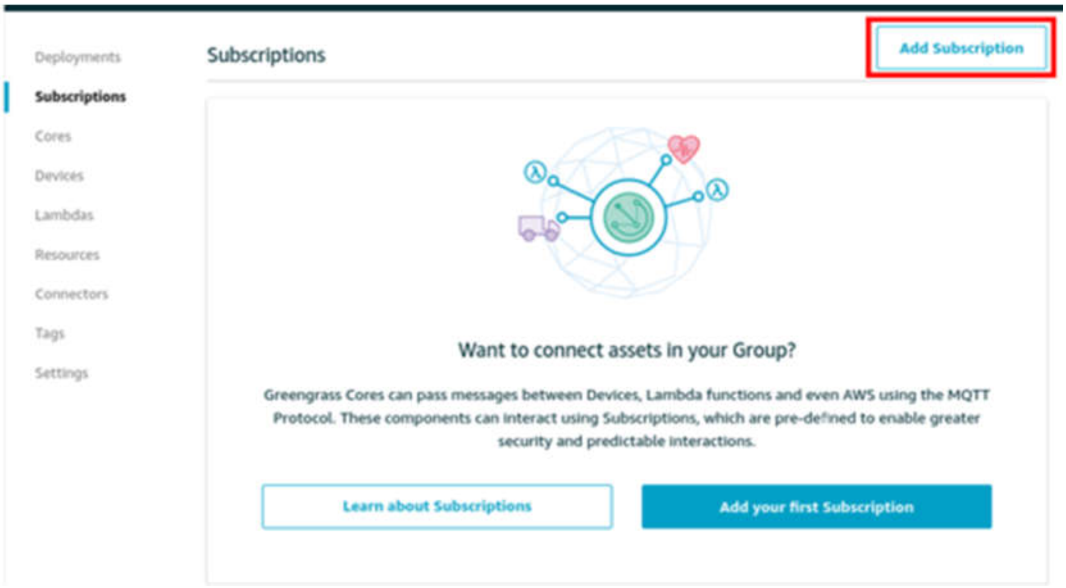


Resources

Name	Resource Type	Status	Local path
gg_ml_resource	Model	● Affiliated	/home/ubuntu/models

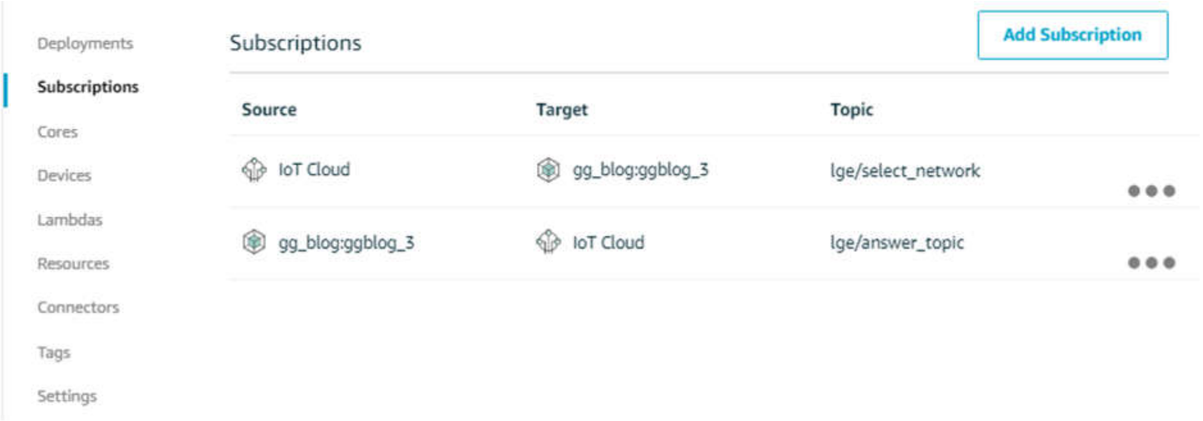
7. Configure Subscription

In the Subscription, source is the name of the Lambda that you registered, and target is the AWS IoT Greengrass, so select IoT Cloud and enter the Topic name.



Want to connect assets in your Group?

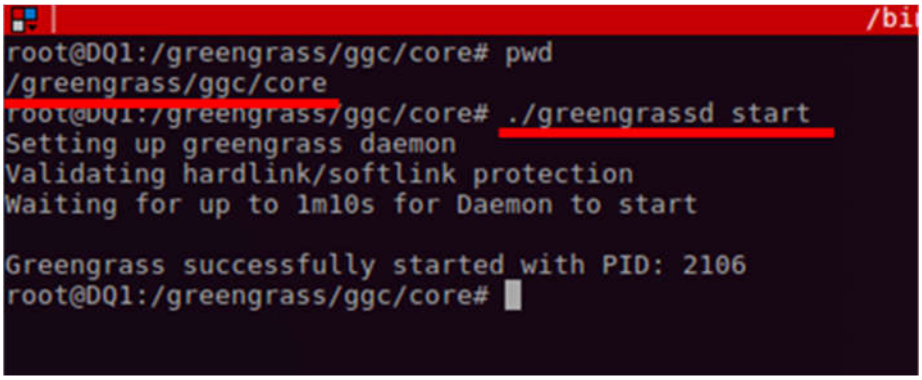
Greengrass Cores can pass messages between Devices, Lambda functions and even AWS using the MQTT Protocol. These components can interact using Subscriptions, which are pre-defined to enable greater security and predictable interactions.



Source	Target	Topic
IoT Cloud	gg_blog:ggblog_3	lge/select_network
gg_blog:ggblog_3	IoT Cloud	lge/answer_topic

8. Deploy AWS IoT Greengrass groups

AWS IoT Greengrass Daemon must be running on the LG AIoT Board before Deploying. Run the AWS IoT Greengrass Daemon with the "greengrass start" command at the "/greengrass/gcc/core" location during installation, and the AWS IoT Greengrass Daemon with root privileges is located at the "/greengrass/gcc/core" location.

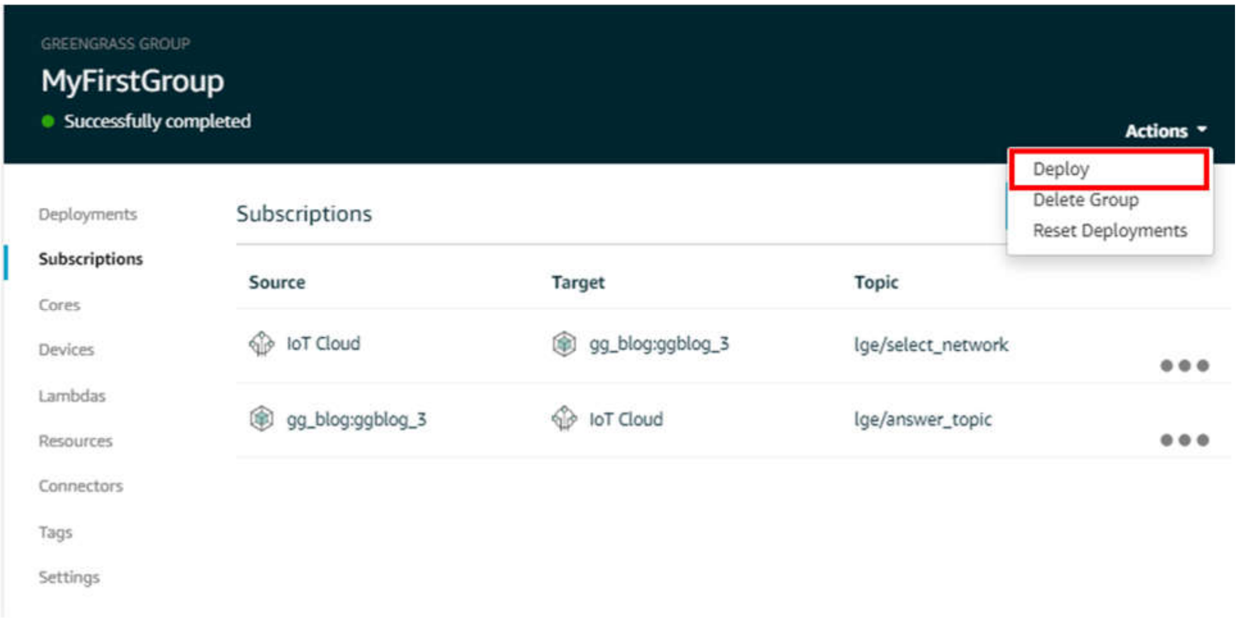


```
root@DQ1:/greengrass/ggc/core# pwd
/greengrass/ggc/core
root@DQ1:/greengrass/ggc/core# ./greengrassd start
Setting up greengrass daemon
Validating hardlink/softlink protection
Waiting for up to 1m10s for Daemon to start

Greengrass successfully started with PID: 2106
root@DQ1:/greengrass/ggc/core# 
```

On AWS IoT Greengrass console, press the Actions button in the upper right corner of the screen, and select Deploy. On the Configure how devices discover your core page, choose Automatic detection. This enables devices to automatically acquire connectivity information for the core, such as IP address, DNS, and port number.

When Deploy is in progress, the circle on the left side of the screen changes from gray to yellow to green, and each signifies preparation, transfer in progress, and deployment is complete.



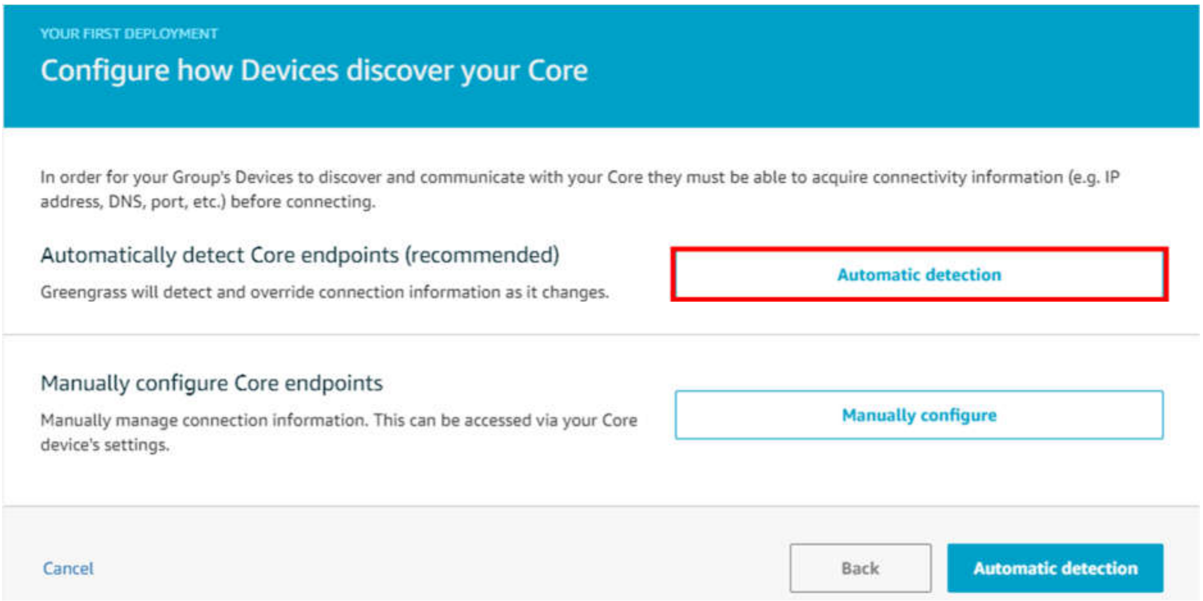
GREENGRASS GROUP

MyFirstGroup

● Successfully completed

Subscriptions

Source	Target	Topic
IoT Cloud	gg_blog:ggblog_3	lge/select_network
gg_blog:ggblog_3	IoT Cloud	lge/answer_topic



YOUR FIRST DEPLOYMENT

Configure how Devices discover your Core

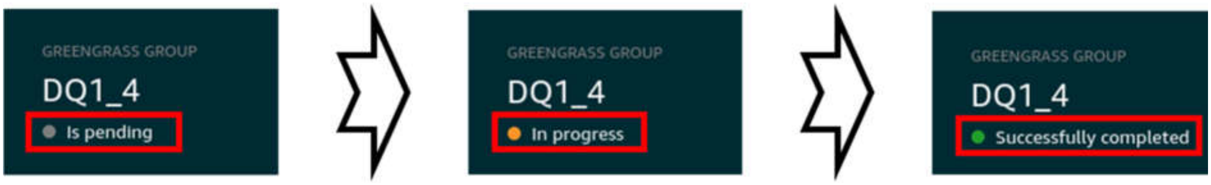
In order for your Group's Devices to discover and communicate with your Core they must be able to acquire connectivity information (e.g. IP address, DNS, port, etc.) before connecting.

Automatically detect Core endpoints (recommended)

Greengrass will detect and override connection information as it changes.

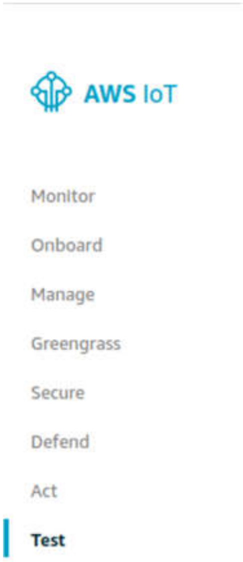
Manually configure Core endpoints

Manually manage connection information. This can be accessed via your Core device's settings.



8. Test

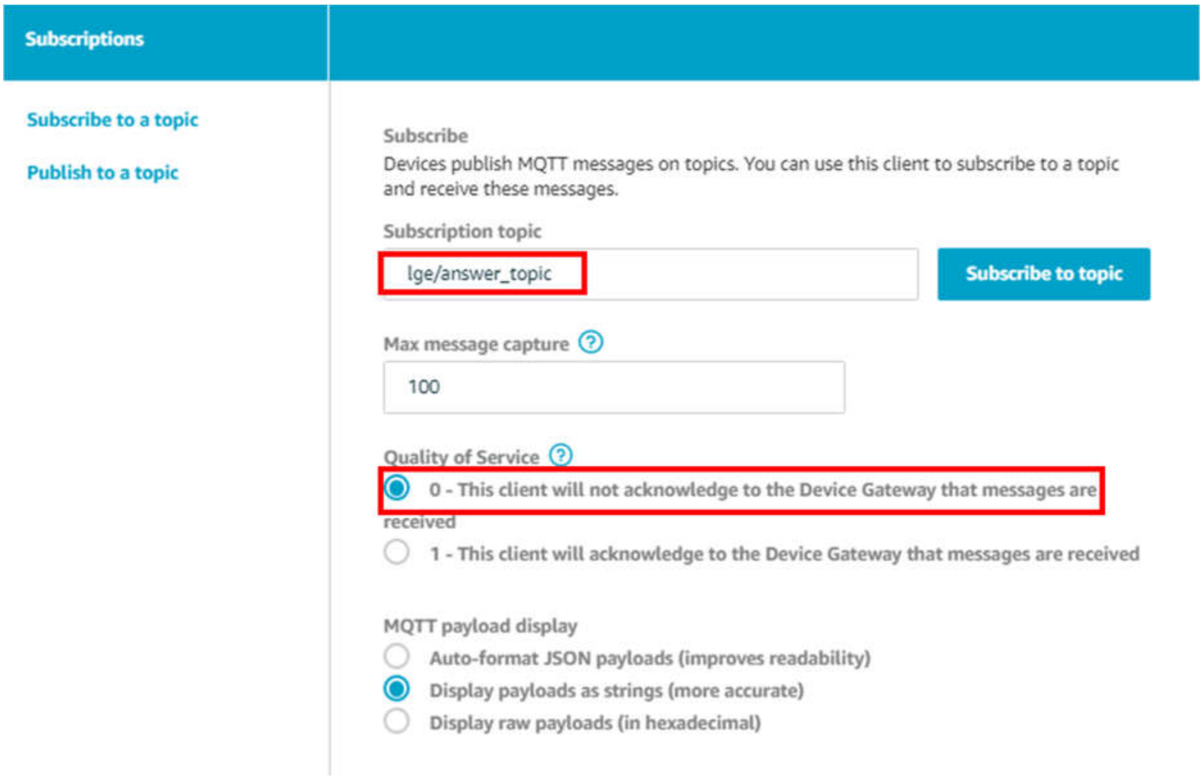
After your deployment is complete, return to the AWS IoT console and choose Test.



For Subscription topic, enter lge/answer_topic to receive local inferencing result.

For Quality of Service, choose 0.

For MQTT payload display, choose Display payloads as strings



Subscribe

Devices publish MQTT messages on topics. You can use this client to subscribe to a topic and receive these messages.

Subscription topic: lge/answer_topic

Max message capture: 100

Quality of Service

● 0 - This client will not acknowledge to the Device Gateway that messages are received

○ 1 - This client will acknowledge to the Device Gateway that messages are received

MQTT payload display

○ Auto-format JSON payloads (improves readability)

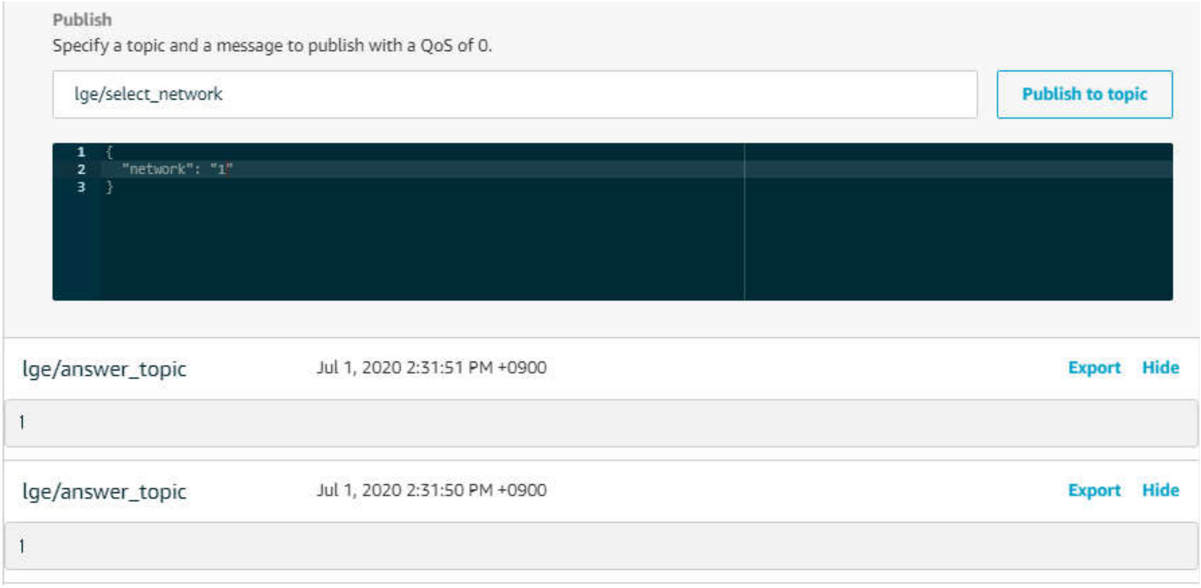
● Display payloads as strings (more accurate)

○ Display raw payloads (in hexadecimal)

For Publish Topic, enter lge/select_network to selct network model.

The number in message payload 1, 2, 3 stands for Mtcnn, MobileNet, and Tiny-Yolo accordingly.

- Mtcnn : number of the face detected
- Tiny-yolo : classified object
- Mobilenet : detected object



Specify a topic and a message to publish with a QoS of 0.

lge/select_network

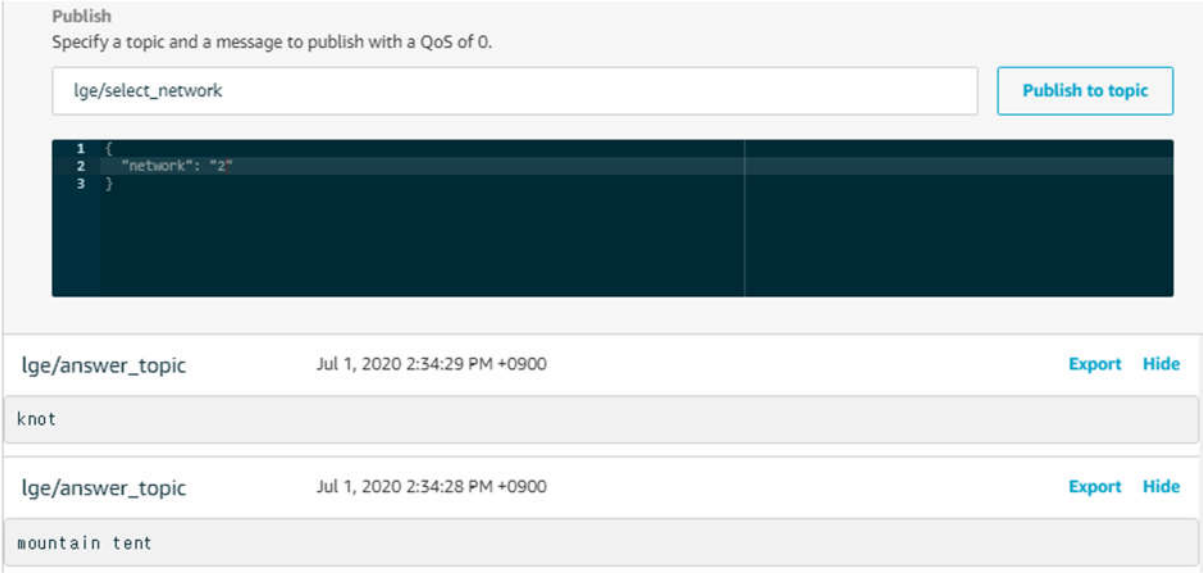
```
{
  "network": "1"
}
```

lge/answer_topic Jul 1, 2020 2:31:51 PM +0900

```
1
```

lge/answer_topic Jul 1, 2020 2:31:50 PM +0900

```
1
```



Specify a topic and a message to publish with a QoS of 0.

lge/select_network

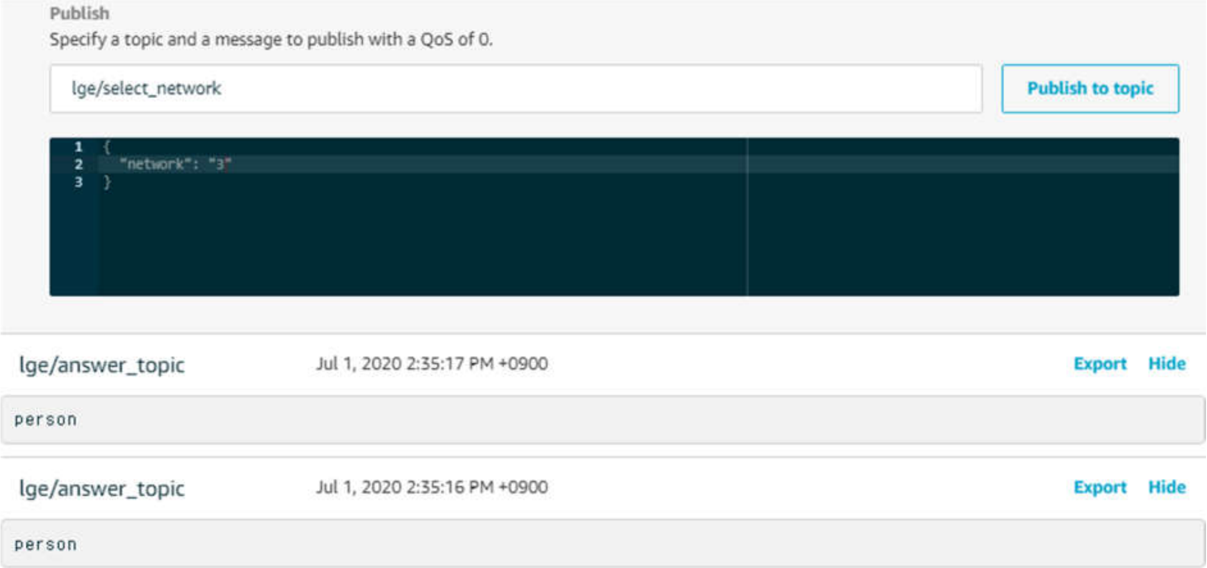
```
{
  "network": "2"
}
```

lge/answer_topic Jul 1, 2020 2:34:29 PM +0900

```
knot
```

lge/answer_topic Jul 1, 2020 2:34:28 PM +0900

```
mountain tent
```



Publish

Specify a topic and a message to publish with a QoS of 0.

lge/select_network

```
{
  "network": "3"
}
```

lge/answer_topic Jul 1, 2020 2:35:17 PM +0900

```
person
```

lge/answer_topic Jul 1, 2020 2:35:16 PM +0900

```
person
```