Experiment: Edge Computing with AWS IoT Greengrass Acceleration

This experiment will start with the basics of AWS Greengrass for Edge & IoT and move to more advanced topics.

During this experiement you will learn how to install the AWS Edge IoT platform AKA Greengrass, connect Edge devices to AWS IoT and Greengrass, and create communication scenarios with those Edge Computing devices and AWS Greengrass.

The experiment will be exercised on an Elastic Compute Cloud (EC2) instance which will be provisioned through AWS CloudFormation. CloudFormation is one of the Infrastructure as Code (IaC) tools in AWS. Amazon Linux is used as operating system and the standard username is ***ec2-user***.

The OpenSource Cloud9 IDE will be installed by you on and EC2 instance. Cloud9 is an integrated development environment (IDE) that lets you write, run, and debug your code with just a browser. It also provides shell access in the browser which is needed for the exercises in this workshop. We will access the IDE through a web browser interface and you will set the username/password for that access. Ensure that you take notes if you change the username and password from the information provided in the experiment. The only way to reset those credentials is to delete our CloudFormation stack and recreate should you forget them.

For the experiment virtual devices are provisioned into your Cloud9 IDE EC2 virtual machine instance. You will find the Python device driver source located in the following files:

- **/home/ec2-user/greengrass-bootcamp/ggad-1/GGBootcampPubSub.py**
- **/home/ec2-user/greengrass-bootcamp/ggad-2/GGBootcampPubSub.py**

The **awscli** is installed and configured on the Cloud9 Integrated Development Environment (IDE). Where the experiment instructions refer to the **awscli** use the one that is installed on the EC2 instance we built with CloudFormation for our Cloud9 IDE.

Edge & IoT Foundation Steps

- Getting Started
- Edge Computing with AWS IoT
- Edge & IoT - AWS Greengrass
- Experiment Architecture
- Launch EC2 Instance with CloudFormation
- Enable logging for AWS IoT
- Connect a thing to AWS IoT

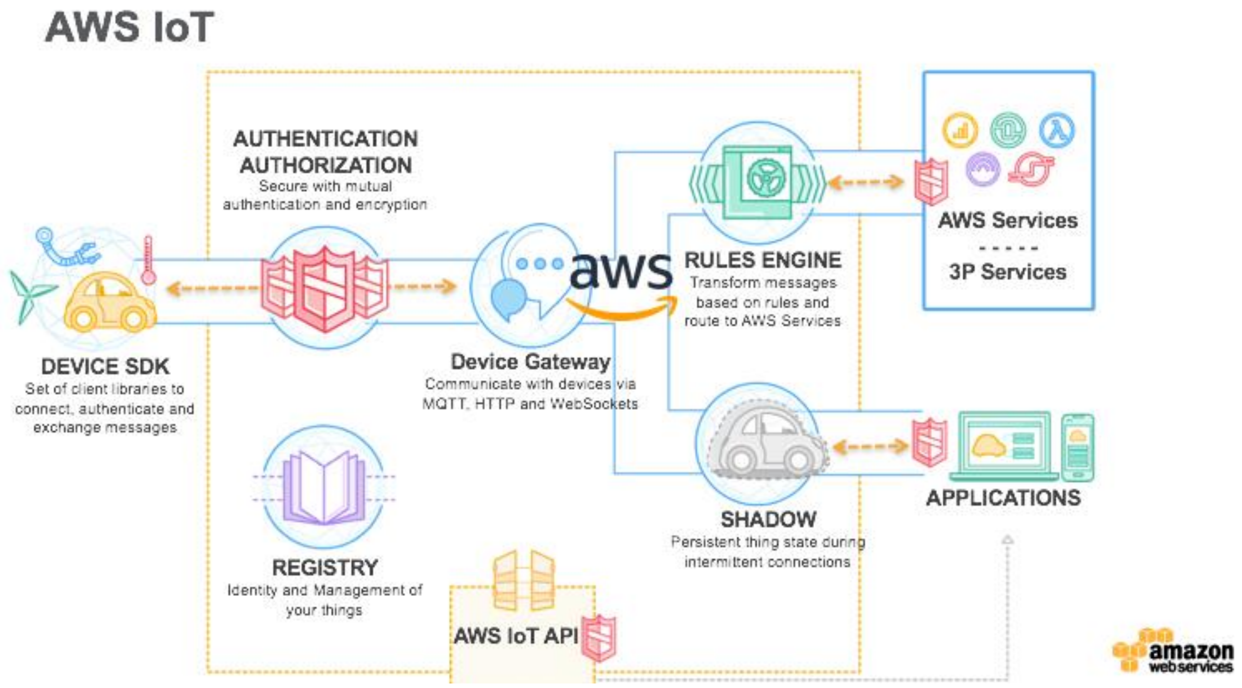
- Send sensor data to AWS IoT
- Provision a Greengrass Group
- Configure Greengrass on your device
- Over The Air Update (OTA)
- Add devices to the Greengrass Group
- Enable Logging for Greengrass
- Deploy the Greengrass Group to the device
- Connect devices to the Greengrass Core
- Device to device communication
- Device to cloud communication
- Cloud to device communication
- Troubleshooting
- Clean Up
- Survey

Prerequisites

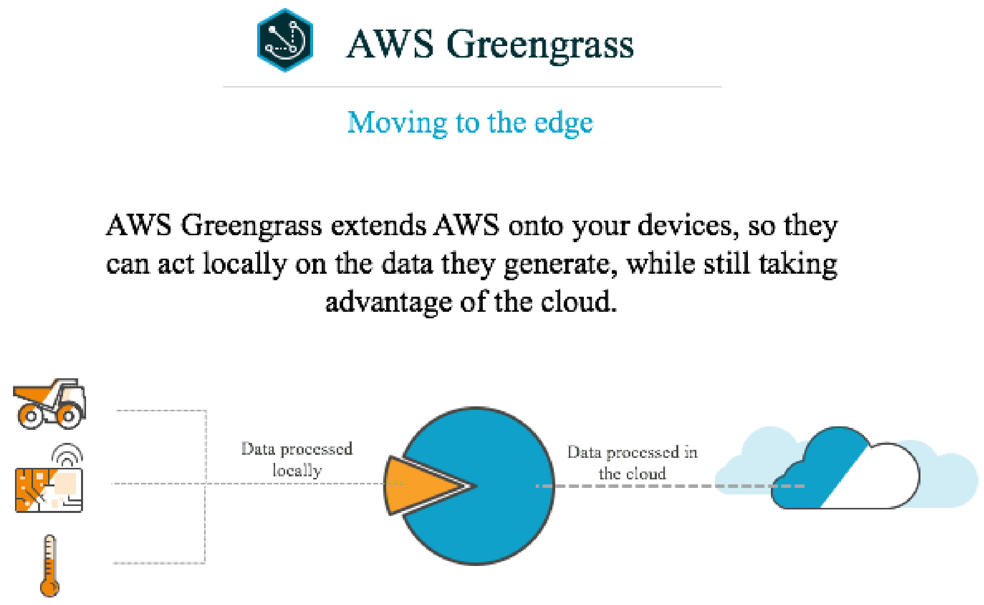
To conduct this experiment, we need:

- AWS Account with admin credentials provided by the instructor
- Your computer (desktop, laptop or tablet)
- Browser (Chrome preferred)
- Basic Linux knowledge

Edge Computing with AWS IoT


AWS IoT
AUTHENTICATION AUTHORIZATION
Secure with mutual authentication and encryption
DEVICE SDK
Set of client libraries to connect, authenticate and exchange messages
aws
Device Gateway
Communicate with devices via MQTT, HTTP and WebSockets
RULES ENGINE
Transform messages based on rules and route to AWS Services
AWS Services
- - - - -
3P Services
SHADOW
Persistent thing state during intermittent connections
APPLICATIONS
REGISTRY
Identity and Management of your things
AWS IoT API
amazon web services

Edge & IoT - AWS Greengrass


AWS Greengrass

Moving to the edge

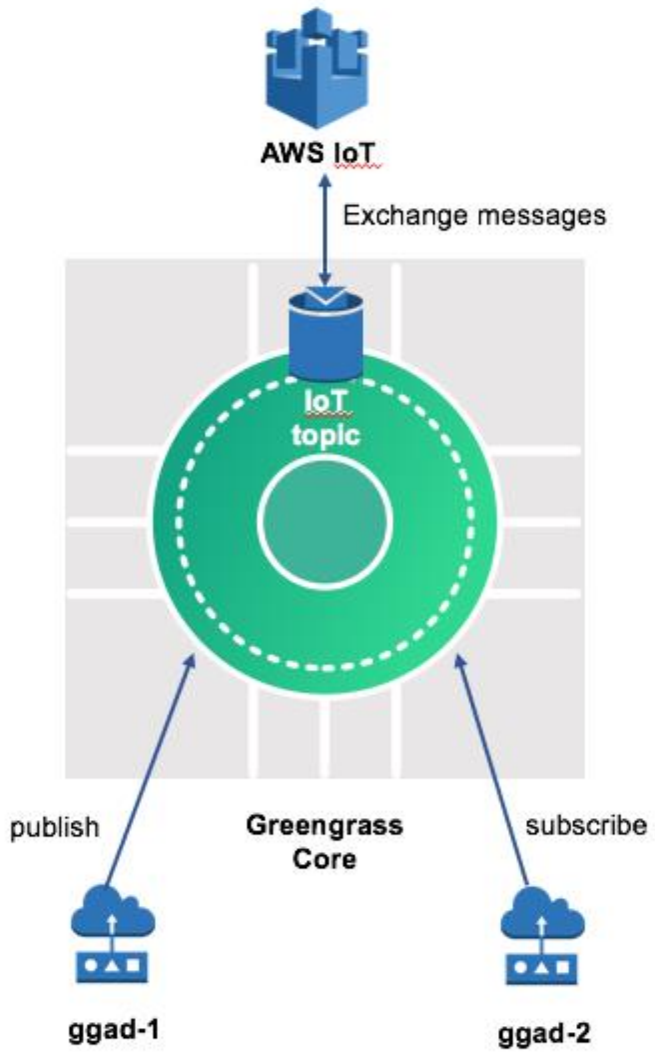
AWS Greengrass extends AWS onto your devices, so they can act locally on the data they generate, while still taking advantage of the cloud.

AWS Greengrass

Features

Experiment Architecture

You will build this architecture:



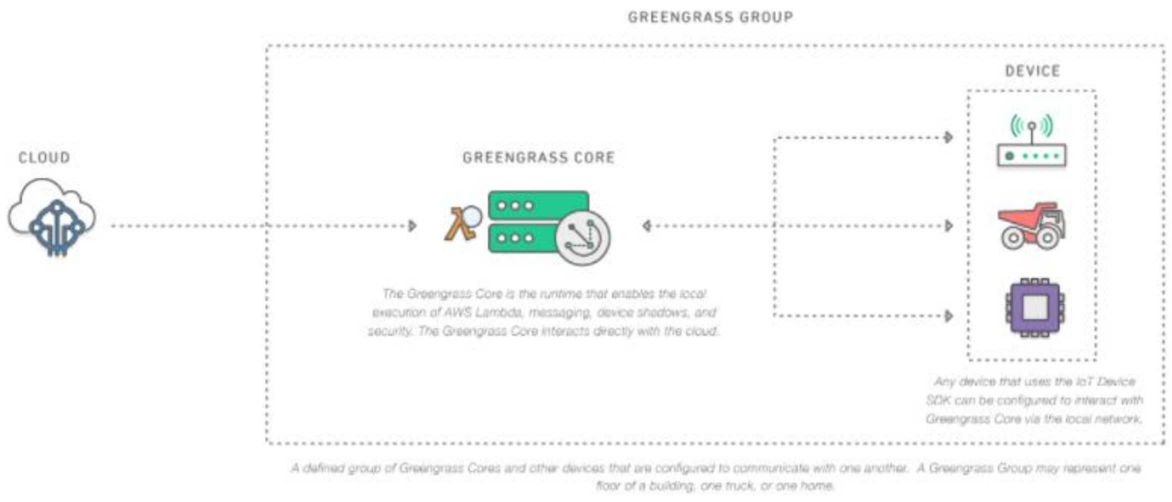
Part 4: Provision a Greengrass Group

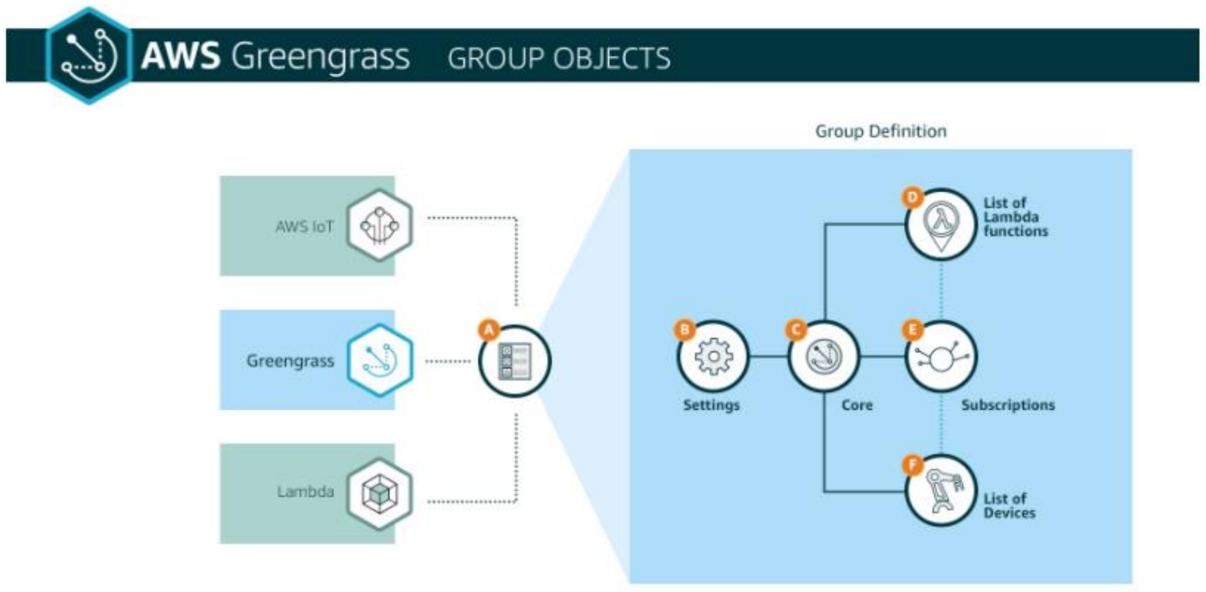
Overview:

In this portion of our experiment, we will get an overview of the group concepts in AWS Greengrass. Afterwards you will create a Greengrass group and install the AWS Greengrass software on you device.

Devices will connect to your Greengrass core and send data locally as well as to the cloud and receive messages from the cloud.

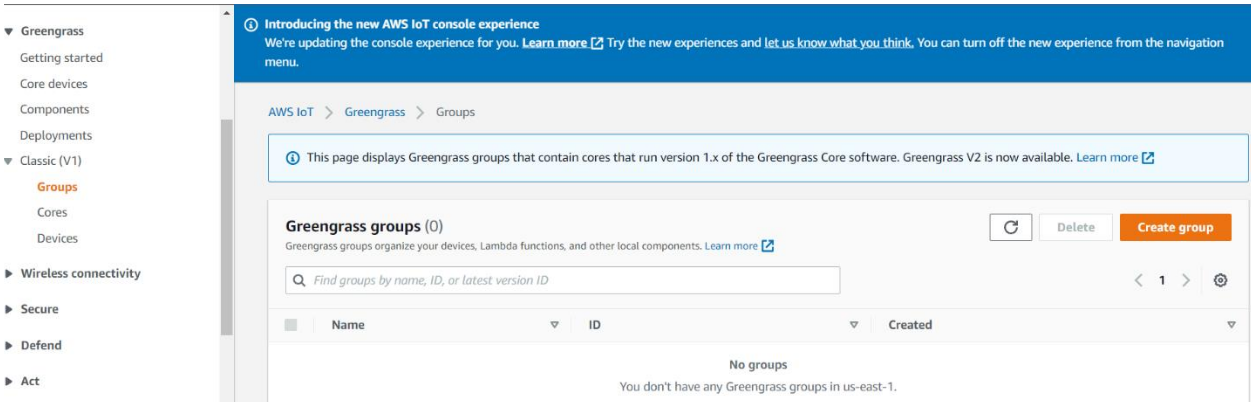
Group Concepts


GREENGRASS GROUP
CLOUD
GREENGRASS CORE
DEVICE
The Greengrass Core is the runtime that enables the local execution of AWS Lambda, messaging, device shadows, and security. The Greengrass Core interacts directly with the cloud.
Any device that uses the IoT Device SDK can be configured to interact with Greengrass Core via the local network.
A defined group of Greengrass Cores and other devices that are configured to communicate with one another. A Greengrass Group may represent one floor of a building, one truck, or one home.



Create a Greengrass group

Go to the AWS Greengrass console - https://console.aws.amazon.com/greengrass/



Introducing the new AWS IoT console experience
We're updating the console experience for you. Learn more Try the new experiences and let us know what you think. You can turn off the new experience from the navigation menu.

AWS IoT > Greengrass > Groups

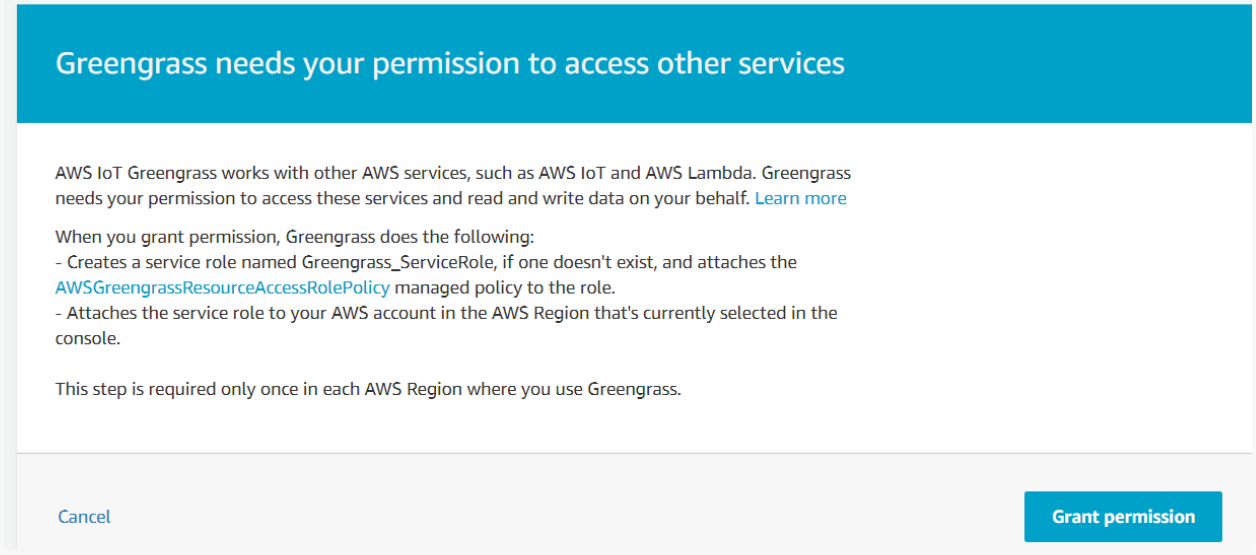
This page displays Greengrass groups that contain cores that run version 1.x of the Greengrass Core software. Greengrass V2 is now available. Learn more

Greengrass groups (0)
Greengrass groups organize your devices, Lambda functions, and other local components. Learn more

Name	ID	Created

No groups
You don't have any Greengrass groups in us-east-1.

Expand Greengrass
Expand Classic (V1)
Select **Groups** under the Class
Select **Create Group**



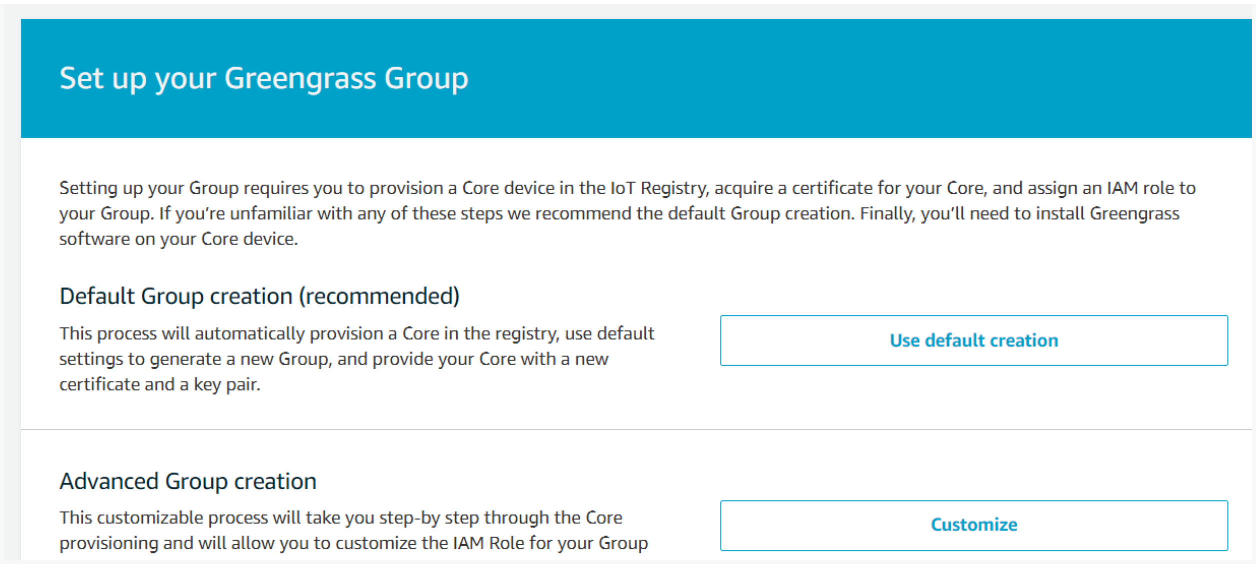
Greengrass needs your permission to access other services

AWS IoT Greengrass works with other AWS services, such as AWS IoT and AWS Lambda. Greengrass needs your permission to access these services and read and write data on your behalf. Learn more

When you grant permission, Greengrass does the following:
- Creates a service role named Greengrass_ServiceRole, if one doesn't exist, and attaches the AWSGreengrassResourceAccessRolePolicy managed policy to the role.
- Attaches the service role to your AWS account in the AWS Region that's currently selected in the console.

This step is required only once in each AWS Region where you use Greengrass.

Select **Grant permission**


Set up your Greengrass Group

Setting up your Group requires you to provision a Core device in the IoT Registry, acquire a certificate for your Core, and assign an IAM role to your Group. If you're unfamiliar with any of these steps we recommend the default Group creation. Finally, you'll need to install Greengrass software on your Core device.

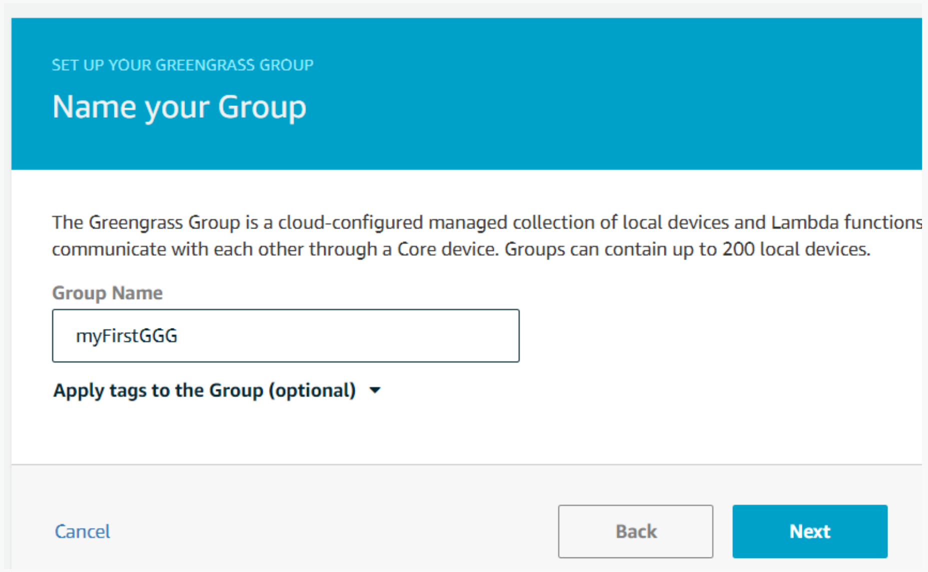
Default Group creation (recommended)

This process will automatically provision a Core in the registry, use default settings to generate a new Group, and provide your Core with a new certificate and a key pair.

Advanced Group creation

This customizable process will take you step-by step through the Core provisioning and will allow you to customize the IAM Role for your Group

Select **Use default creation**

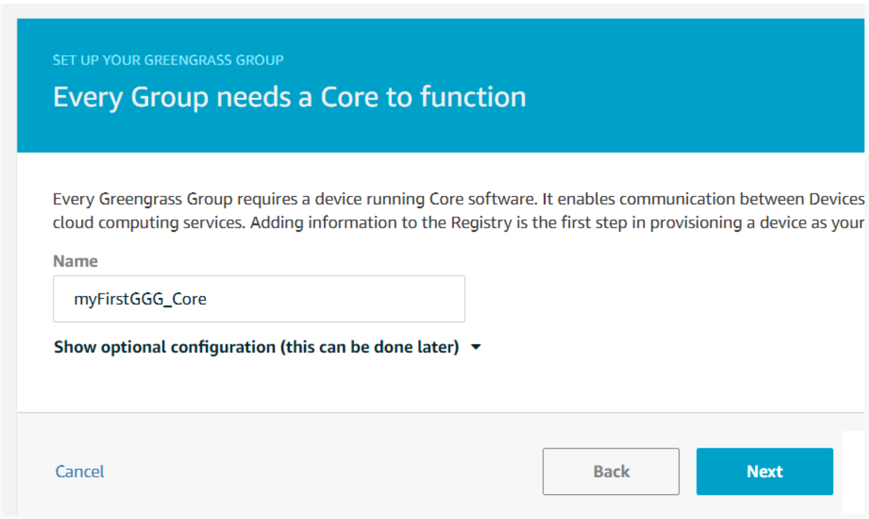

SET UP YOUR GREENGRASS GROUP

Name your Group

The Greengrass Group is a cloud-configured managed collection of local devices and Lambda functions communicate with each other through a Core device. Groups can contain up to 200 local devices.

Group Name: myFirstGGG

Enter Group Name: **myFirstGGG**
Select **Next**



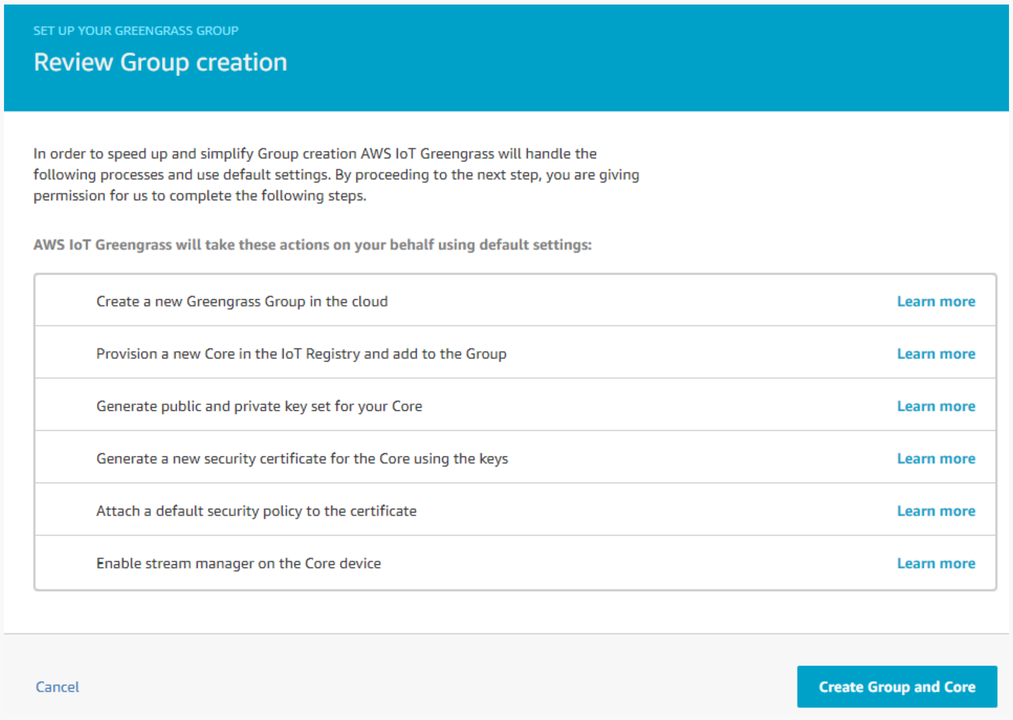
SET UP YOUR GREENGRASS GROUP

Every Group needs a Core to function

Every Greengrass Group requires a device running Core software. It enables communication between Devices
cloud computing services. Adding information to the Registry is the first step in provisioning a device as your

Name: myFirstGGG_Core

Leave Name for Core defaulted to the value presented
Select **Next**



SET UP YOUR GREENGRASS GROUP

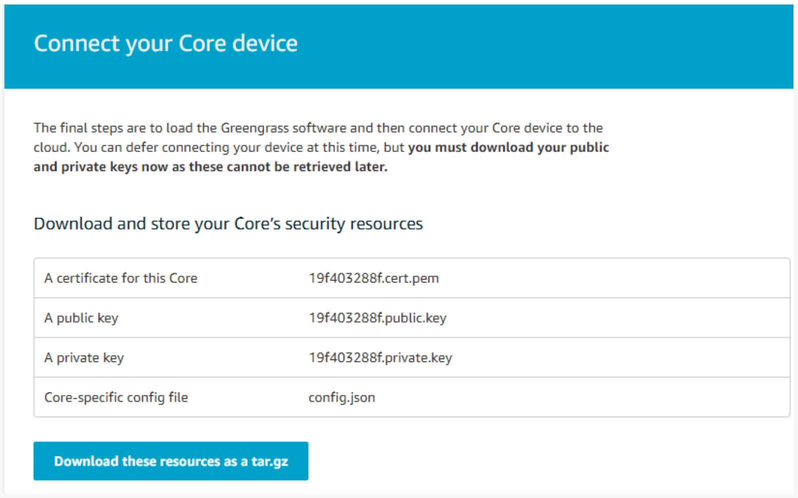
Review Group creation

In order to speed up and simplify Group creation AWS IoT Greengrass will handle the following processes and use default settings. By proceeding to the next step, you are giving permission for us to complete the following steps.

AWS IoT Greengrass will take these actions on your behalf using default settings:

Create a new Greengrass Group in the cloud	Learn more
Provision a new Core in the IoT Registry and add to the Group	Learn more
Generate public and private key set for your Core	Learn more
Generate a new security certificate for the Core using the keys	Learn more
Attach a default security policy to the certificate	Learn more
Enable stream manager on the Core device	Learn more

Select **Create Group and Core**


Connect your Core device

The final steps are to load the Greengrass software and then connect your Core device to the cloud. You can defer connecting your device at this time, but **you must download your public and private keys now as these cannot be retrieved later.**

Download and store your Core's security resources

A certificate for this Core	19f403288f.cert.pem
A public key	19f403288f.public.key
A private key	19f403288f.private.key
Core-specific config file	config.json

Select **Download these resources as a tar.gz**

If you were NOT using the EC2 instance created by CloudFormation, we would have to download the Greengrass software, and the Root CA, as well
Select **Finish**

Note: !!! Don't forget to click "Finish". Otherwise your group will not be created !!!

Verify in the AWS Greengrass console that your Greengrass Group has been created

Greengrass -> Groups

Configure Greengrass on your device

To configure AWS Greengrass so that the software is ready to start you need to create a service role and copy and unpack the tar.gz-file that you downloaded earlier onto your EC2 instance.

The Greengrass software is already copied and unpacked on your EC2 instance.

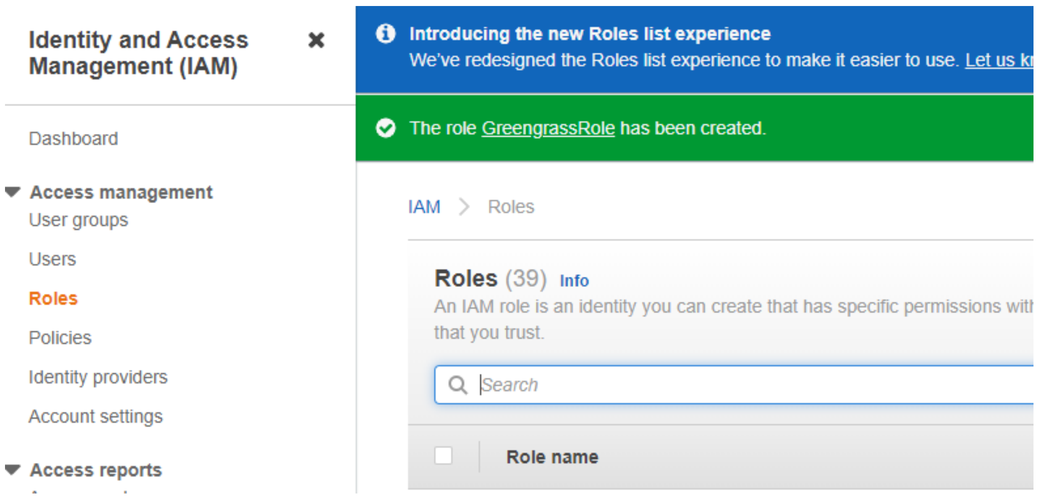
Create a Greengrass Service Role

AWS Greengrass requires access to your AWS Lambda and AWS IoT data. Therefor you must create the service role.

Go to the AWS IAM console

Select **Roles** in the left navigation pane
Select **Create role**
Leave default selection as **AWS service**
Select **Greengrass** and there is a single use case which will be selected by default
Next: Permissions

Select **AWSGreengrassResourceAccessRolePolicy**
Select **Next: Tags**
Enter **project** for the tag key and for the tag value enter **edge-computing-iot-acceleration**
Select **Next: Review**
Enter Role name: **GreengrassRole**
Select **Create role**


Identity and Access Management (IAM)

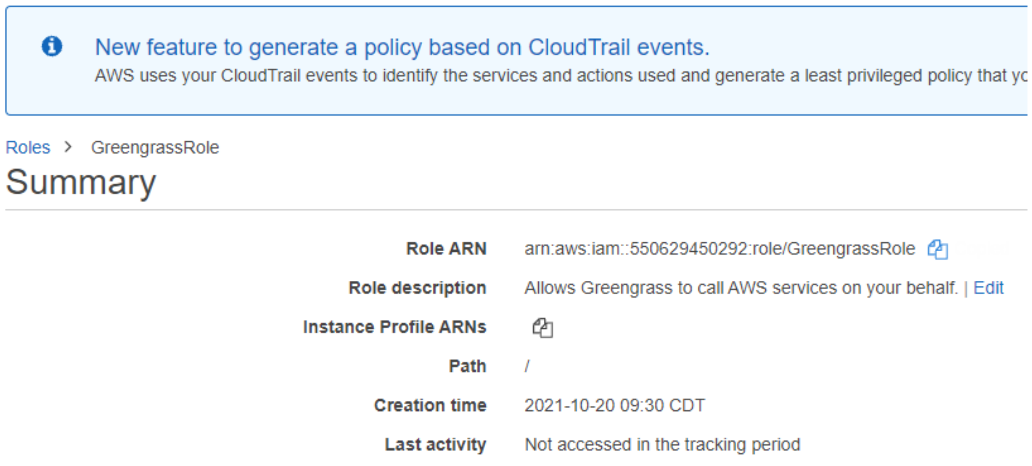
Introducing the new Roles list experience
We've redesigned the Roles list experience to make it easier to use. Let us k

The role GreengrassRole has been created.

IAM > Roles

Roles (39) Info
An IAM role is an identity you can create that has specific permissions wit
that you trust.

Select the GreengrassRole hyperlink in the created message.


New feature to generate a policy based on CloudTrail events.
AWS uses your CloudTrail events to identify the services and actions used and generate a least privileged policy that yo

Roles > GreengrassRole

Summary

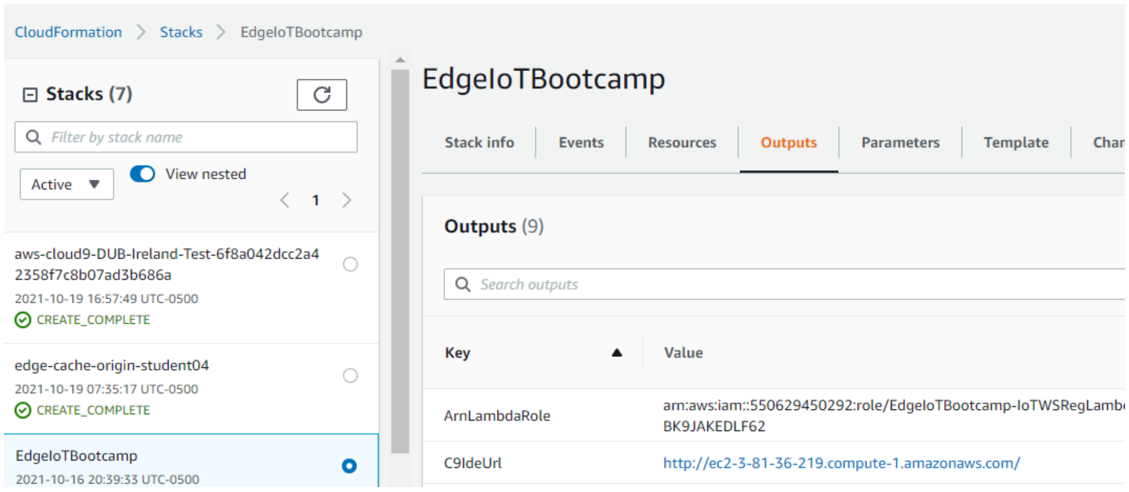
Role ARN	arn:aws:iam::550629450292:role/GreengrassRole
Role description	Allows Greengrass to call AWS services on your behalf. | Edit
Instance Profile ARNs	
Path	/
Creation time	2021-10-20 09:30 CDT
Last activity	Not accessed in the tracking period

Copy the **Role ARN** and use it to create your awscli command.

The Role ARN should look like arn:aws:iam::550629450292:role/GreengrassRole

- In a Cloud9 terminal:

Remember that we can use the CloudFormation console and view the outputs for our **EdgeIoTBootcamp** stack to retrieve the Cloud9 IDE URL (C9IdeUrl) from the Foundation experiment


CloudFormation > Stacks > EdgeIoTBootcamp

Stacks (7)

Filter by stack name

Active ▼ View nested 1

aws-cloud9-DUB-Ireland-Test-6f8a042dcc2a42358f7c8b07ad3b686a
2021-10-19 16:57:49 UTC-0500
CREATE_COMPLETE

edge-cache-origin-student04
2021-10-19 07:35:17 UTC-0500
CREATE_COMPLETE

EdgeIoTBootcamp
2021-10-16 20:39:33 UTC-0500

EdgeIoTBootcamp

Stack info | Events | Resources | Outputs | Parameters | Template | Char

Outputs (9)

Search outputs

Key	Value
ArnLambdaRole	arn:aws:iam::550629450292:role/EdgeIoTBootcamp-IoTWSRegLamb BK9JAKEDLF62
C9IdeUrl	http://ec2-3-81-36-219.compute-1.amazonaws.com/

```
aws greengrass associate-service-role-to-account --role-arn
arn:aws:iam::<YOUR_AWS_ACCOUNT_ID>:role/GreengrassRole
```

You can find the role's arn in the summary section for the role.

Prepare Over The Air Update (OTA) - Optional

This portion of the experiment is optional If you decide not to do it go to ***Copy and unpack the tar.gz-file***

The AWS Greengrass core software comes packaged with an OTA Update Agent that is capable of updating the core's software or the OTA Update Agent itself to the latest respective versions. You can start an update by invoking the CreateSoftwareUpdateJob API or from the Greengrass console.

To use OTA for Greengrass two Greengrass software versions have been copied onto the EC2 instances:

- Software in **/greengrass** is the current version
- Software in **/greengrass-X.Y.Z** is the previous version. X.Y.Z is a placeholder for the Greengrass version number, e.g. 1.5.0

For testing OTA the previous Greengrass software must be moved to **/greengrass**

In a Cloud9 terminal:

```
cd /
sudo mv greengrass greengrass-current
sudo mv greengrass-X.Y.Z greengrass
```

Continue with the next step.

Copy and unpack the tar.gz-file (Not optional)

- Copy the downloaded tar.gz-file onto your EC2 instance. The tar.gz file's name is similar to -setup.tar.gz
- The tar.gz file contains keys, certificate and a configuration file (config.json) which will be used to configure your Greengrass Core.
- In a Cloud9 terminal:

  ```
  sudo tar zxvf <unique-string>-setup.tar.gz -C /greengrass/
  ```
- Now you are ready to start your Greengrass core.

But before you start the Greengrass daemon subscribe to the following topics. If the Core starts correctly you can observe activities on that topics.

Go to the AWS IoT Core console

```
1. Test
2. Subscribe $aws/events/#, $aws/things/# and #
```

Now fire up Greengrass on your EC2 instance:

- In a Cloud9 terminal:

```
• cd /greengrass/ggc/core
```

  ```
  sudo ./greengrassd start
  ```

Look at the MQTT client in the AWS IoT console for output. You need to become **root** to access the log-directories on the Greengrass Core:

```
sudo su -
```

In a Cloud9 terminal:

```
cd /greengrass/ggc/var/log/system/
tail -f runtime.log
```

If there are any problems when starting AWS Greengrass check file "crash.log" for errors:

```
/greengrass/ggc/var/log/crash.log
```

Your AWS Greengrass Core should be up and running.

Over The Air Update (OTA) - Optional

In this exercise you will learn how to update the Greengrass Core software over the air. You must have followed the optional steps to prepare OTA.

To initiate an OTA update a job must be created. This update job for the Greengrass Core software will be created by the awscli.

To create the awscli command you need to gather the following information:

- [YOUR_CORE_ARN]: The arn of your Greengrass core. You can find this information either:
    - On the EC2 instance in the file **/greengrass/config/config.json**
    - In the AWS Greengrass Console -> Groups -> `myFirstGGG` -> Cores -> `myFirstGGG_Core` -> Details
    - In a Cloud9 terminal: `aws iot describe-thing --thing-name myFirstGGG_Core`
- [S3_URL_SIGNER_ROLE_ARN]: the required role was created automatically by CloudFormation. The role arn can be found in the outputs section of you CloudFormation stack.

1. Subscribe in the AWS IoT Console to the topic **$aws/events/#**
2. In a Cloud9 terminal:
3. Start the ota agent

```
4. cd /greengrass/ota/ota_agent
```

```
sudo ./ggc-ota
```

5. Verify that the ota agent is running

```
6. # is the ota_agent process running?
7. ps aux | grep ggc-ota | grep -v grep
8.
9. # take a look at the log file
```

```
cat /var/log/greengrass/ota/ggc_ota.txt
```

10. Tail the log file from the ota agent

```
tail -f /var/log/greengrass/ota/ggc_ota.txt
```

11. Create the update job

```
12. aws greengrass create-software-update-job \
13.     --update-targets-architecture x86_64 \
14.     --update-targets [YOUR_CORE_ARN] \
15.     --update-targets-operating-system amazon_linux \
16.     --software-to-update core \
17.     --s3-url-signer-role [S3_URL_SIGNER_ROLE_ARN] \
18.     --update-agent-log-level INFO \
```

```
    --amzn-client-token myClientToken1
```

19. Watch for messages in the AWS IoT Console

20. Verify if the updated Greengrass Core software is running. In the process list you should find the version.

```
ps aux |grep greengrass
```

21. The output of the command should contain a line similar to: "/greengrass/ggc/packages/**1.5.0_1**/bin/daemon -core-dir=/greengrass/ggc/packages/**1.5.0_1** -greengrassdPid=30149" In this example you can see that a Greengrass version **1.5.0** is running.

Add devices to the Greengrass Group

The Greengrass Group consists currently only of a Core. In this step you will add devices to the Group. You will use the devices "ggad-1" and "ggad-2" which you used before. But these device will later be configured to connect to your Core instead of AWS IoT.

Go to the AWS Greengrass console

```
Select Groups
Select myFirstGGG
Select Devices
Select Add your first Device
Select an IoT Thing
Select ggad-1 -> Finish
Repeat to Add ggad-2 in the same way to your Greengrass Group
```

Create a subscription

The two devices ggad-1 and ggad-2 which you just added to the Greengrass Group should communicate where ggad-1 acts as a publisher and ggad-2 as a subscriber.

To route messages between devices and therefore allow communication a so called subscription must be defined.

A subscription is a routing rule which consists of a source, a target an a topic filter. A subscription defines which source may communicate to which target on which topic.

Go to the AWS Greengrass console

```
Select Groups
Select myFirstGGG
Select Subscriptions
Select Add your first Subscription
Select a source -> Devices -> ggad-1
  Select a target -> Devices -> ggad-2
 Select Next
Optional topic filter -> sdk/test/Python
  Select Next
Select  Finish
```

We have now created devices and subscriptions in the Greengrass Group and are ready for the first deployment. But before deploying for the first time let's enable logging for Greengrass.

Enable Logging for Greengrass

By default logging is not enabled for the Greengrass Core. Logging should be enabled to get insights what happens on the Core and also for troubleshooting purposes.

Go to the AWS Greengrass console

```
Select Groups
Select myFirstGGGSelect Settings
Scroll down
   Local logs configuration -> Edit
Add another log type
Check both "User Lambdas" and "Greengrass system"
   Update
What level of logs should be sent? -> Select Debug logs
 Select Save
```

Logfiles on the Greengrass Core:

```
Log directory: /greengrass/ggc/var/log
System logs: /greengrass/ggc/var/log/system
Lambda logs: /greengrass/ggc/var/log/user/<AWS_REGION>/<ACCOUNT_ID>
```

Deploy the Greengrass Group to the device

After a Greengrass Group was created or changed the configuration needs to be deployed to the Greengrass Core.

In a Cloud9 terminal:

```
cd /greengrass/ggc/var/log/system
tail -f localwatch/localwatch.log *.log
```

Go to the AWS Greengrass console

```
1. Select Groups
2. Select myFirstGGG
3. Actions -> Deploy
4. Select Automatic detection
```

After some moments you should see activity in the logs on the device and the deployment status at the console.

If your deployment should fail with an error regarding missing permission you probably didn't setup the service role for Greengrass correctly.

Connect devices to the Greengrass Core

Now that the Greengrass core got a configuration set through a deployment devices will be connected to the Greengrass Core.

To connect devices to a Greengrass Core the connectivity information - ip address/DNS name and port - is required as well as the CA certificate which signed the Core's certificate.

These settings can be obtained automatically by devices through the discovery service in AWS IoT. To use the discovery service an IoT policy is required which allows the action **greengrass:Discover**.

To connect the devices ggad-1 and ggad-2 to the core by using the connectivity information from the discovery service the action **greengrass:Discover** must be added to the policy of each device. The policy is also in the GitHub repo as GreengrassPolicy.json

Go to the AWS IoT Core console

```
1. Manage
2. Things
2. Choose ggad-1
3. Security
4. Click the certificate
```

```
5. Policies
6. Click ggad-1-Policy
7. Edit policy document
8. in the Action section add "greengrass:Discover" -> Save as new version
```

Your new policy document should look like the following:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iot:Publish",
        "iot:Subscribe",
        "iot:Connect",
        "iot:Receive",
        "greengrass:Discover"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

Change the policy document also for the device "ggad-2"

The Greengrass Core can have multiple Core endpoints that means connection information. The script that you will use to connect to the GG Core will iterate through the connections advertised by the core and will use the first successful connection.

Take a look at the available Core endpoints:

Go to the AWS IoT Core console

1. Select **Cores**
2. Select **myFirstGGG_Core**
3. Select **Connectivity**

Device to device communication

The devices ggad-1 and ggad-2 will connect to the Greengrass Core and communicate locally. ggad-1 will act as publisher and ggad-2 as a subscriber which will write the data received from ggad-1 to STDOUT.

By default the devices are connecting to AWS IoT. To point them to a Greengrass Core the command line parameter "--connect-to greengrass" is used.

Modify for **both** devices the script "start.sh" and add "--connect-to greengrass" to the line which starts the Python script GGBootcampPubSub.py. The resulting line should look like:

```
python GGBootcampPubSub.py -e
<YOUR_ENDPOINT>.<AWS_REGION>.amazonaws.com -r root-CA.crt -c ggad-
1.cert.pem -k ggad-1.private.key --clientId ggad-1 --connect-to greengrass
```

Open two terminals and start the devices with the script start.sh:

```
Terminal 1: start ggad-2
Terminal 2: start ggad-1
Terminal 1: you should see messages arriving from ggad-1
```

Information regarding message routing can be found in the log file `/greengrass/ggc/var/log/system/runtime.log`.

Device to cloud communication

In this exercise messages will be send from a device (ggad-2) to the cloud. To route the messages accordingly from a device to the cloud another subscription has to be created.

Create a subscription in your Greengrass Group with:

```
Source: Device ggad-2
Target: IoT Cloud
Optional topic filter: sdk/test/Python
Deploy
```

Go to the AWS IoT Core console

```
Subscribe in AWS IoT to the topic sdk/test/Python
```

In a Cloud9 terminal:

```
Start ggad-2 from the ggad-2 folder
$> sudo ./start.sh
```

Look for incoming messages in the AWS IoT MQTT client

Cloud to device communication

It is also possible to send messages from the cloud to a GGAD. In this exercise data from the cloud should be routed to the device ggad-1. As you may guess another subscription and deployment is needed.

Create a subscription in your Greengrass Group with:

```
Source: IoT Cloud
Target: Device ggad-1
Optional topic filter: sdk/test/Python
Deploy
```

In a Cloud9 terminal:

```
Start ggad-1
```

Go to the AWS IoT MQTT client and publish a message to sdk/test/Python

```
1. Publish to a topic
2. Publish: topic: sdk/test/Python
   You can leave the default message untouched
3. Publish to topic
```

On your device in the window/terminal where ggad-1 is running look for incoming messages.

Troubleshooting

To troubleshoot your environment you can ssh into the EC2 instance. This could be necessary e.g. when the IDE fails to start.

While the EC2 instance is bootstrapped an ssh key-pair is generated. A S3 Bucket is also created as part of your working environment. The private ssh-key is stored in the S3 Bucket as *ssh/iotws*. The private key is protected with the same password used for the Cloud9 IDE.

You can find the bucket name in the outputs section of your CloudFormation stack.

Clean Up

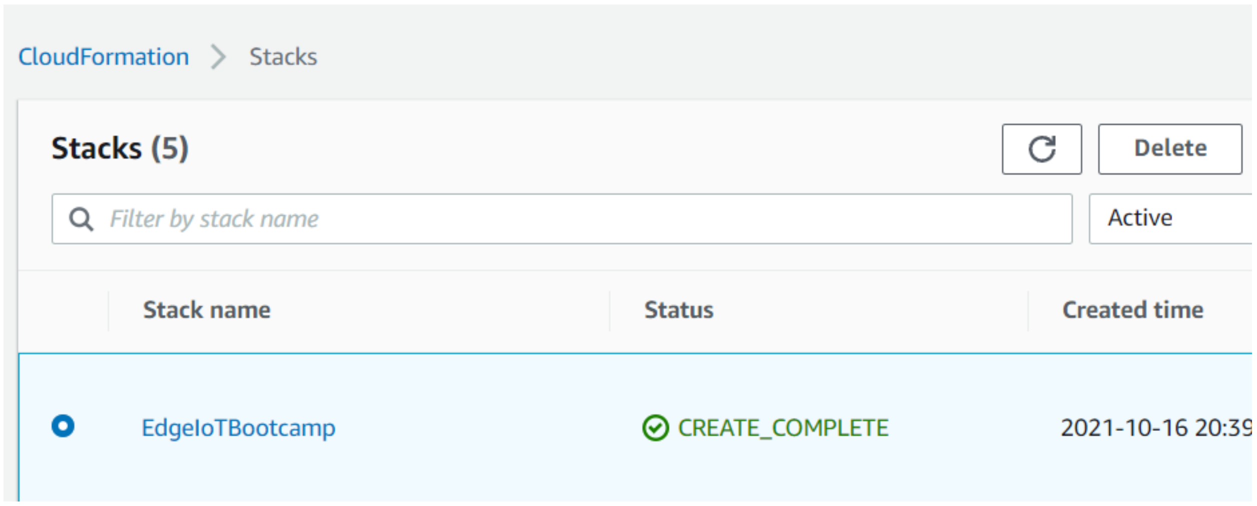
- Empty the S3 Bucket otherwise the CloudFormation stack will fail when deleting the bucket

  In a Cloud9 terminal:

```
aws s3 rm s3://$S3_BUCKET --recursive
```

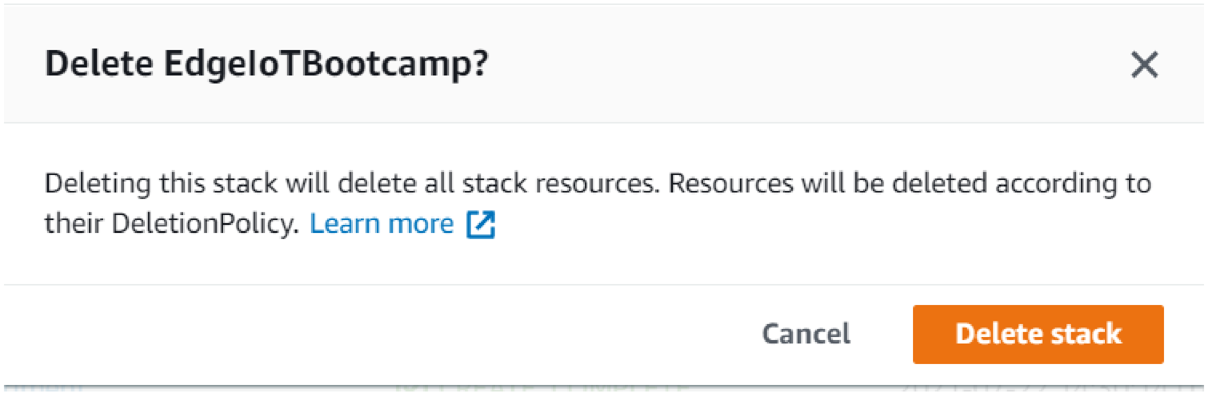
- Delete the CloudFormation stack

  Go to the [AWS CloudFormation console]()



Stack name	Status	Created time
EdgeIoTBootcamp	CREATE_COMPLETE	2021-10-16 20:39

  1. Check the radio for `EdgeIoTBootcamp`
  2. Select **Delete** button



Delete EdgeIoTBootcamp?

Deleting this stack will delete all stack resources. Resources will be deleted according to their DeletionPolicy. Learn more

  3. Select **Delete Stack**

- Delete the Greengrass Group

  Go to the [AWS Greengrass console]()

  1. Groups
  2. myFirstGGG
  3. Actions
  4. Reset Deployments
  5. Reset deployment
  6. Actions
  7. Delete Group
  8. Yes, continue with delete

- Delete the Greengrass core in the IoT device registry

  Go to the [AWS IoT Core console]

  1. Manage
  2. Click `myFirstGGG_Core`
  3. Security
  4. Click the certificate name
  5. Actions
  6. Delete
  7. Yes, continue with delete
  8. Manage
  9. Click `...` at myFirstGGG_Core
  10. Delete
  11. Yes, continue with delete
  12. Secure
  13. Policies
  14. Click `...` at myFirstGGG_Core-policy
  15. Delete
  16. Yes, continue with delete

- Delete IAM role for Greengrass

  Go to the [AWS IAM console]

  1. Roles
  2. Type `GreengrassRole` in the search field
  3. Check `GreengrassRole`
  4. Delete role
  5. Yes, delete