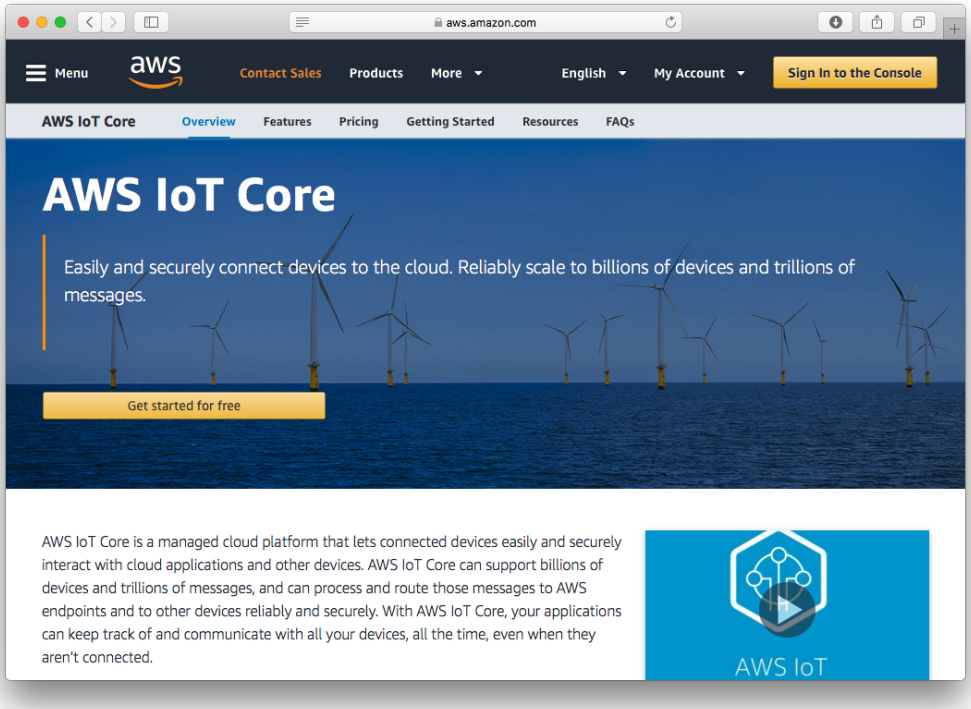



# User Guide

*Connecting your EpiSensor Gateway to AWS IoT Core*

*Applies to: NGR-30-3, NGR-30-5, ASAVIE-SKV1*

EPI-101-01

# Table of Contents

# Introduction

EpiSensor's Internet of Things platform is easy to deploy, configure and scale and includes a range of sensor products that can monitor a variety of environmental and energy usage parameters in commercial and industrial environments.

This user guide contains technical information on how to connect your EpiSensor Gateway to AWS IoT Core, the managed IoT Cloud platform from Amazon Web Services. AWS IoT Core can be used to securely manage the flow of data between your EpiSensor Gateway, and other services for data storage, analysis and visualisation.

This guide requires a minimum of version V04.00.01.00.00 of EpiSensor Gateway software.

## Related Documents

Related installation and configuration documents are listed in the following table:

| Document | Reference No. |
|---|---|
| EpiSensor NGR-30-3 Datasheet | EPI-102-00 |
| EpiSensor NGR-30-5 Datasheet | EPI-077-00 |
| Gateway API User Guide | EPI-009-08 |
| User Guide for NGR | EPI-075-00 |

## Getting Started

There are currently two versions of the EpiSensor Gateway available (NGR-30-3 and NGR-30-5) based on different hardware platforms. The items included with each are as follows:

| NGR-30-3 | | |
|---|---|---|
| | **Qty** | **Item** |
|  | 1 | NGR-30-3 Gateway |
| | 1 | Mains Power Supply |
| | 1 | Ethernet Cable |
| | 1 | 2.4GHz Antenna for ZigBee |

| NGR-30-5 | | |
|---|---|---|
|  | **Qty** | **Item** |
| | 1 | NGR-30-5 Gateway |
| | 1 | 2.4GHz Antenna for ZigBee |
| | 1 | Cellular Antenna |
| | 1 | WLAN Antenna |
| | 1 | Ethernet Cable |

The NGR-30-5 Gateway (based on the Dell Edge Gateway 3002) requires an external 12/24V power supply that is not included as standard, unless you're using one of our starter/accelerator kits. For more information on Dell Edge Gateway 3002 hardware, click here to access the user manual and here for a spec sheet.

You will need an EpiSensor Gateway with an Internet connection (Cellular, Ethernet or Wi-Fi) with TCP port 8883 open, and at least one EpiSensor wireless sensor to get data flowing to AWS IoT Core.

## Configure AWS IoT Core

AWS IoT Core is a managed cloud platform that lets connected devices easily and securely interact with cloud applications. AWS IoT Core can support billions of devices and trillions of messages, and can process and route those messages to AWS endpoints and to other devices and services reliably and securely.

Data can be transferred to AWS IoT Core in a number of ways, but in this guide we'll focus on sending data in JSON format via MQTT (Message Queuing Telemetry Transport), which is an efficient publish-subscribe-based messaging protocol optonised for high-latency, low-bandwidth networks connections.

### *Log in to your AWS Console*

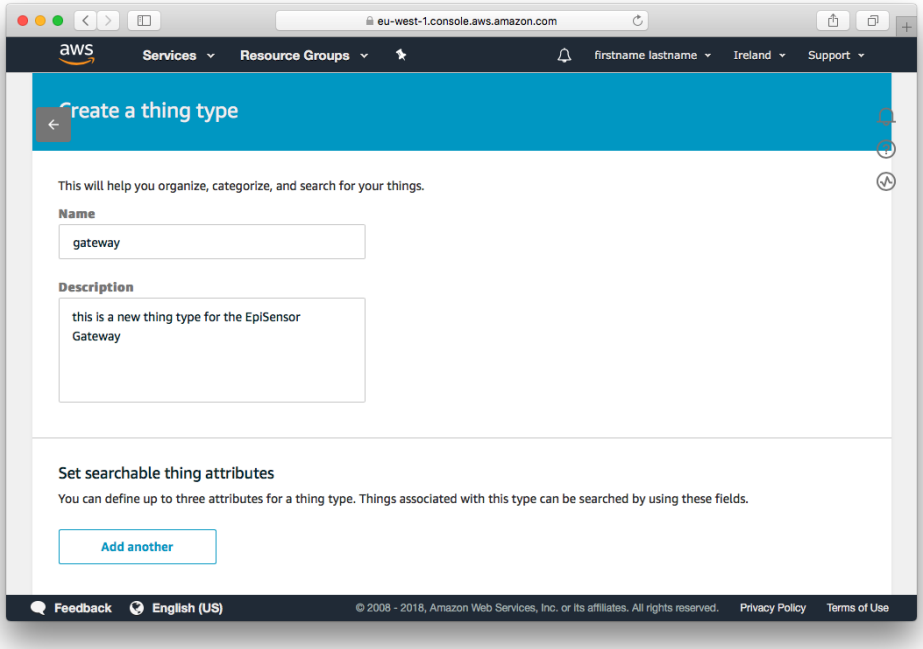
To get started, sign up for an account on AWS or log in to your existing account at the following link: https://console.aws.amazon.com/

We'll first need to configure AWS IoT Core to accept connections from our EpiSensor Gateway. AWS IoT Core refers to devices that connect to the platform as 'Things'.

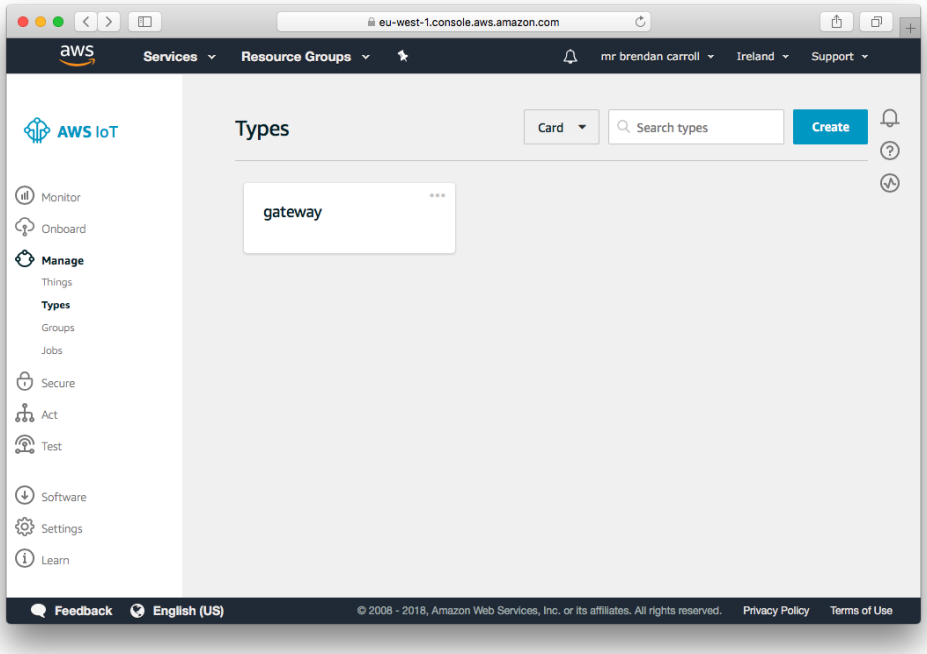
## *Create a new 'Thing type'*

A 'Thing type' lets you define attributes that our new 'Thing' should have. In this case, we'll define a new Thing type called "gateway" and give it an attribute called "serial", which we'll use to record the unique Serial Number of the EpiSensor Gateway. This is optional, but will make it easier to filter our list of 'Things' in the future.

In your AWS console, go to Services → AWS IoT → Manage (in the left navigation bar) → Types. Click the "Create" button and set the name to "gateway" and give it a description.

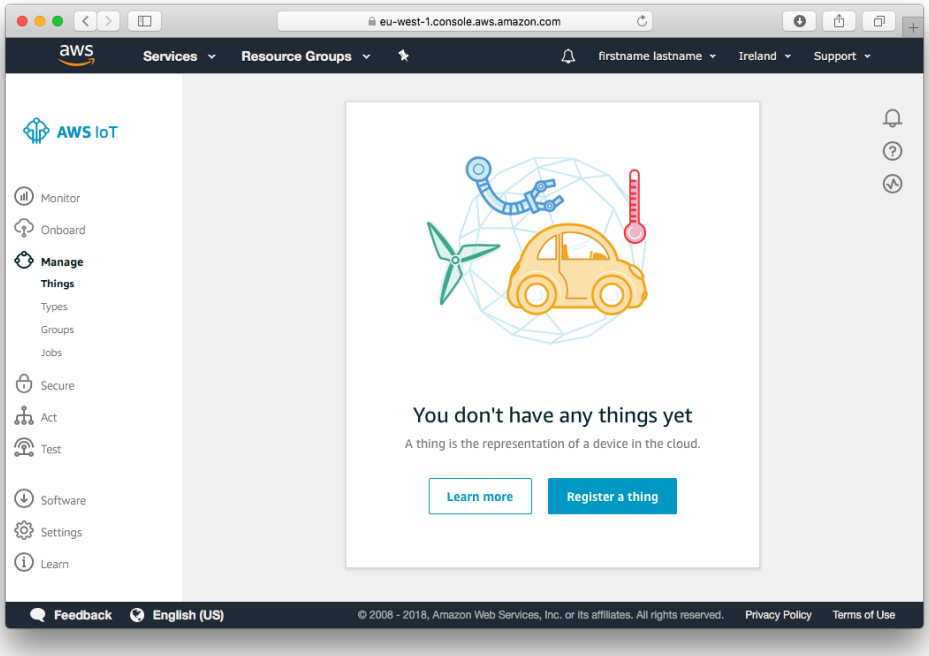



Scroll down the page and configure an Attribute Key called "serial" then click "Create thing type". You should now see a new thing type called "gateway" in the Types list, as shown in the screenshot below:

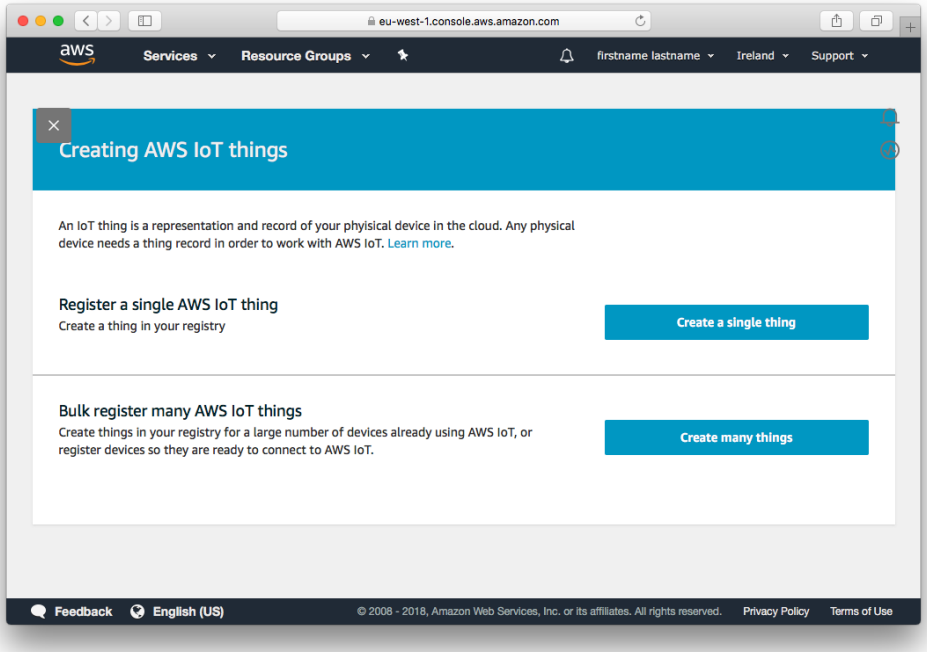



## *Create a new 'Thing'*

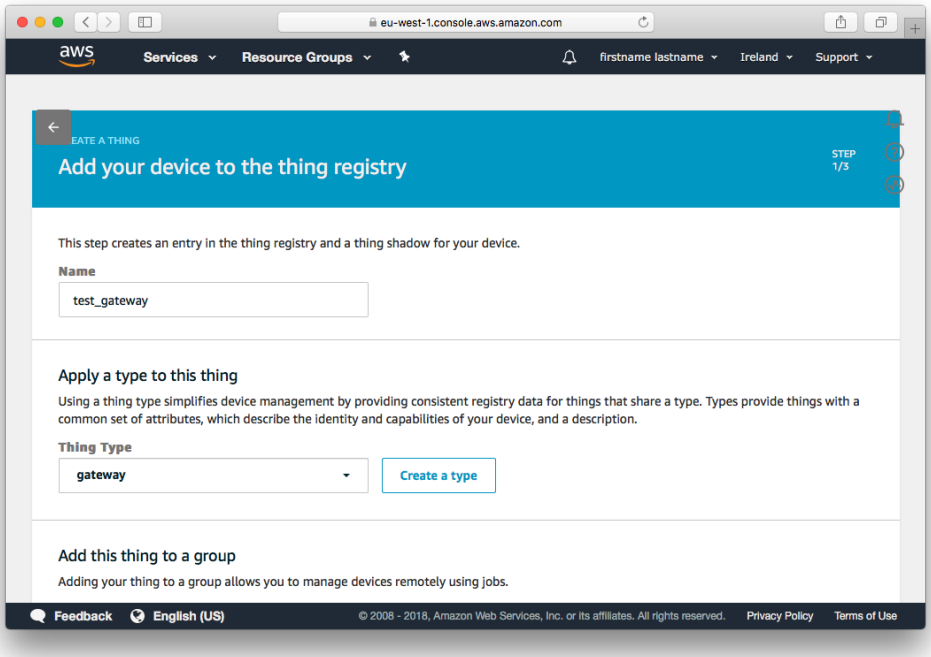
Next, we'll create a new 'Thing' and assign the type we created in the previous step to it. Click on Manage → Things, then click "Create" (or if this it the first time you're creating a Thing, click "Register a Thing" in the middle of the page:



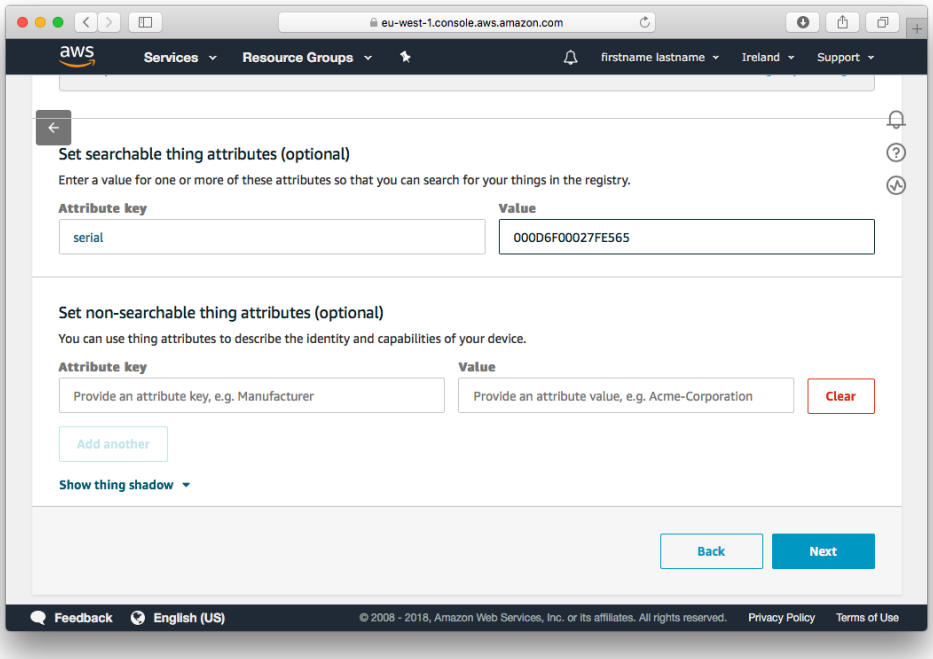
On the next screen, click “Create a single thing”.




Give the new Thing a name and select “gateway” as the type.



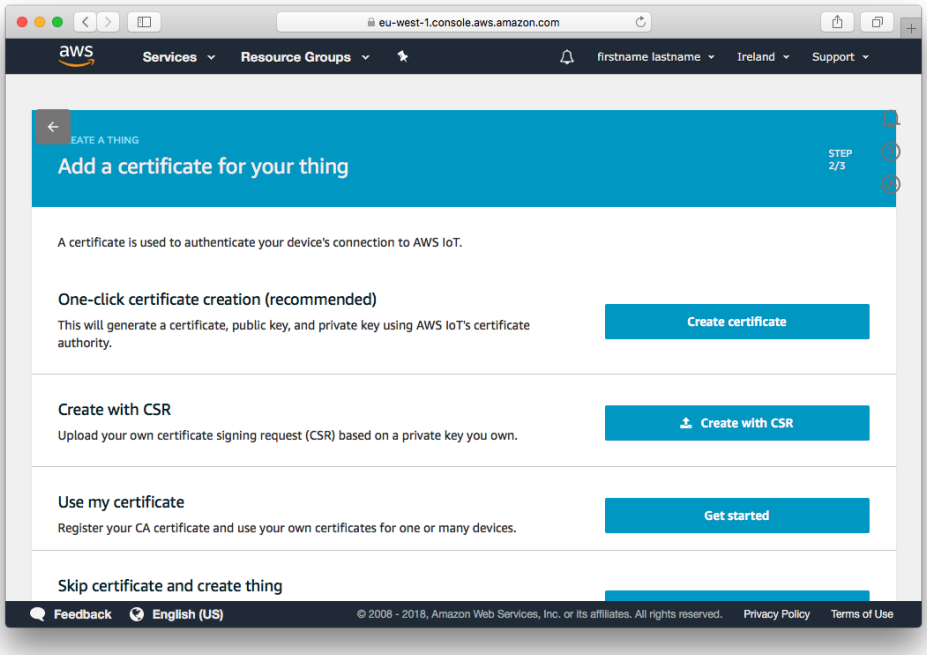

Check the Serial Number of your EpiSensor Gateway from the About → Overview page of the Gateway’s web interface, and set this as an attribute, then click “Next” to move to the next step.



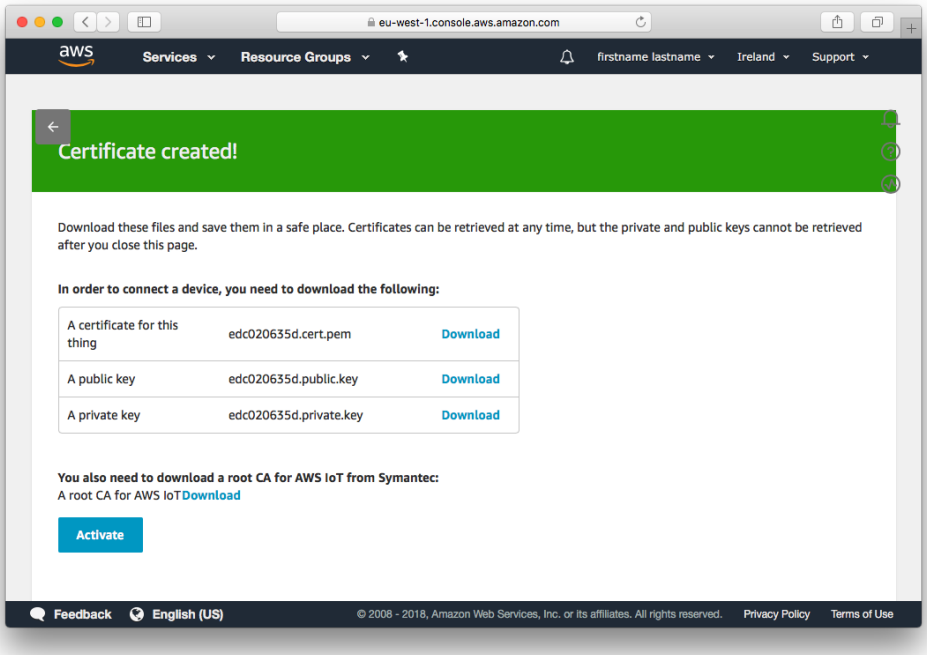

The next step is to create a certificate bundle for our new 'Thing'.

## *Create a Certificate*

On the page below, click "Create certificate". It's possible to use an existing certificates based on a private key, but this is beyond the scope of our guide.

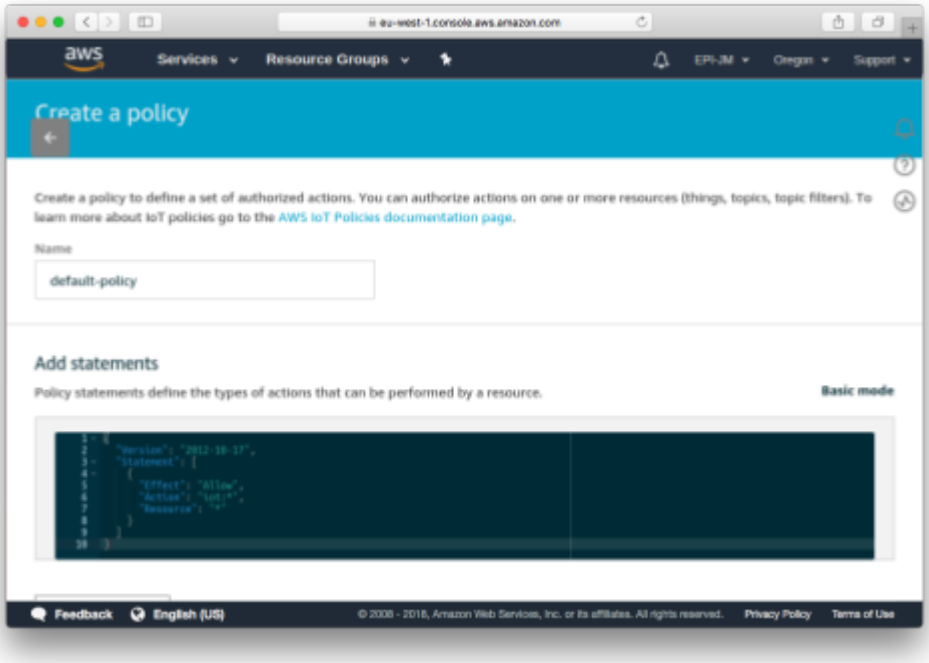



When the security certificate has been created, download the files (including the root CA cert) and click "Activate".



## *Attach a Policy*

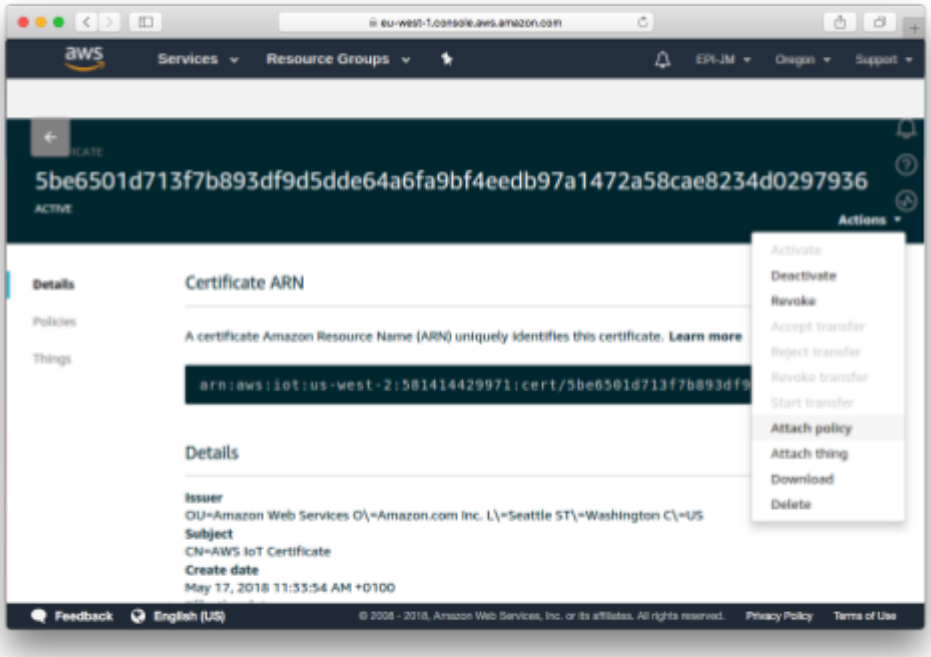
Policies are used to define the permissions that each 'Thing' should have to access AWS IoT Core and other services. For this guide, we'll use a policy that gives our new 'Thing' unrestricted access to AWS IoT Core. The policy should be first created by going to Secure -> Policies -> Create from the main menu on the left.




Name the policy "default-policy" or similar, and click on the "Advanced Mode" link. Paste the following JSON policy into the textbox to allow Things using this policy full access to all resources:

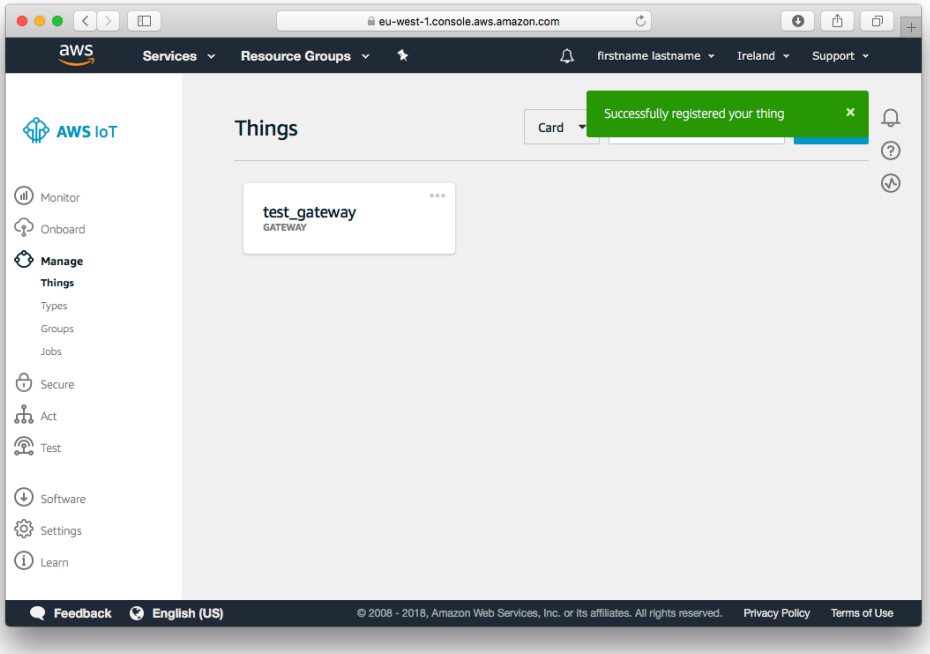
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:*",
      "Resource": "*"
    }
  ]
}
```

Click "Create" to save the policy, then go to Secure > Certificates. Select the certificate we created earlier, and from the Actions menu, click "Attach Policy".




Select “default-policy” from the list, and save. The created policy should now be attached to the security certificate.

After completing these steps, our Thing called “gateway” should be ready for use.



## Configure your EpiSensor Gateway

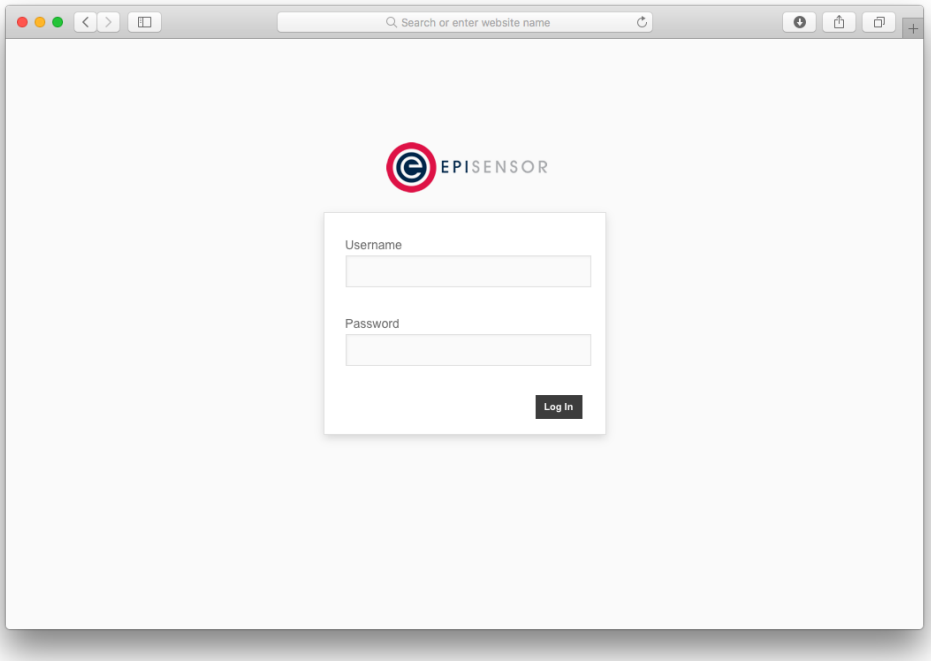
In this section, we'll join a node to the EpiSensor Gateway, configure it to 'export' data, and upload the AWS IoT security certificates so a secure MQTT connection can be established.

### *Logging in*

To log in to your EpiSensor Gateway, first check if the SKU of your hardware is NGR-30-3 or NGR-30-5, as the procedure is slightly different depending on the model.

| Important Note | |
|---|---|
| ⓘ | The factory default IP address of the NGR-30-3 Gateway is 172.31.255.1 |

It can take up to 5 minutes for the Gateway initialise. After this time, go to http://172.31.255.1:8081/ in a modern browser and you should see the Gateway login interface below.




The Gateway supports all recent versions of Internet Explorer (IE 9 +), Chrome, Firefox and Safari web browsers. On older browsers, some features may not display correctly.

| Important Note |
| --- |
| The default Gateway user account is Administrator; the default login details for this account are as follows:<br><br>**Username:** Administrator<br>**Password:** A1 |

If an incorrect password is entered more than four time, users will be locked out for five minutes. After five minutes has elapsed you can try again, up to another five attempts are allowed, and so on. In the event of the password being irretrievable, please contact support@episensor.com

## *Joining a Node*

This section of the guide covers the four steps that are required to connect a node to the Gateway and get data flowing. This assumes that you are using a factory default EpiSensor Gateway, not a kit or a system that has already been in use.

### Forming a Network

The ZigBee network on the Gateway should be 'Formed' once. This will force it to scan for the best wireless channel to operate on, and configure unique encryption keys so the sensors can communicate securely with the Gateway.

To Form a new network, go to Settings → System → Form New Network and click "Submit".

| Important Note |
| --- |
| "Form New Network" should only be done once for every Gateway. Sending this command on a Gateway that already has nodes joined will erase the security keys, and all of the nodes on that Gateway will need to be factory reset and rejoined. |

After about 30 seconds, the scan will have completed, and the Status field on the Home page should show "OK".

### Enable 'Allow Join' Mode

Next, we'll temporarily enable "Allow Join" mode which tells the Gateway to permit new nodes to connect to the Gateway. To do this, go to Settings → Add Nodes, and enable this mode for 15 minutes.

## Waking up your Nodes

If you are joining a battery powered node to the Gateway (assuming it's in factory default condition) you'll need to wake it up from deep sleep mode by pressing and holding the Mode button until the status LED flickers, and then release.

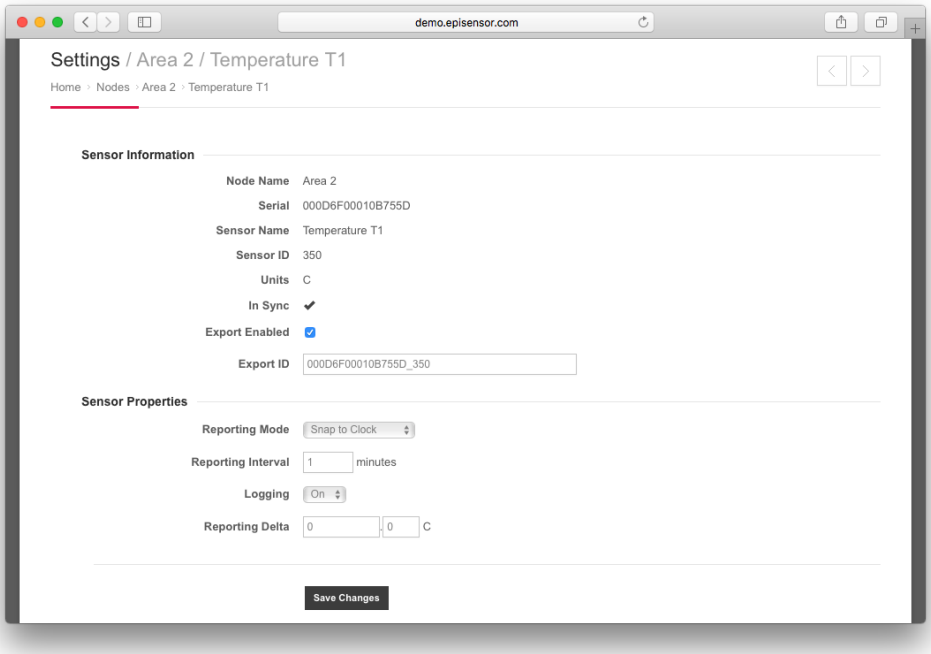
If you're joining a mains-powered node, it should automatically join if it's in factory default condition. To factory reset a node, refer to the node's user manual.

## Enable data export

When you have joined a node to your Gateway, it will need a couple of minutes to synchronise its settings. On the 'Nodes' page, click on Action → Settings. Towards the bottom of this page, you should see a list of the 'sensors' available on the node.

Select one that you are interested in, and click Action → Settings to go a level deeper into the settings of that particular sensor.

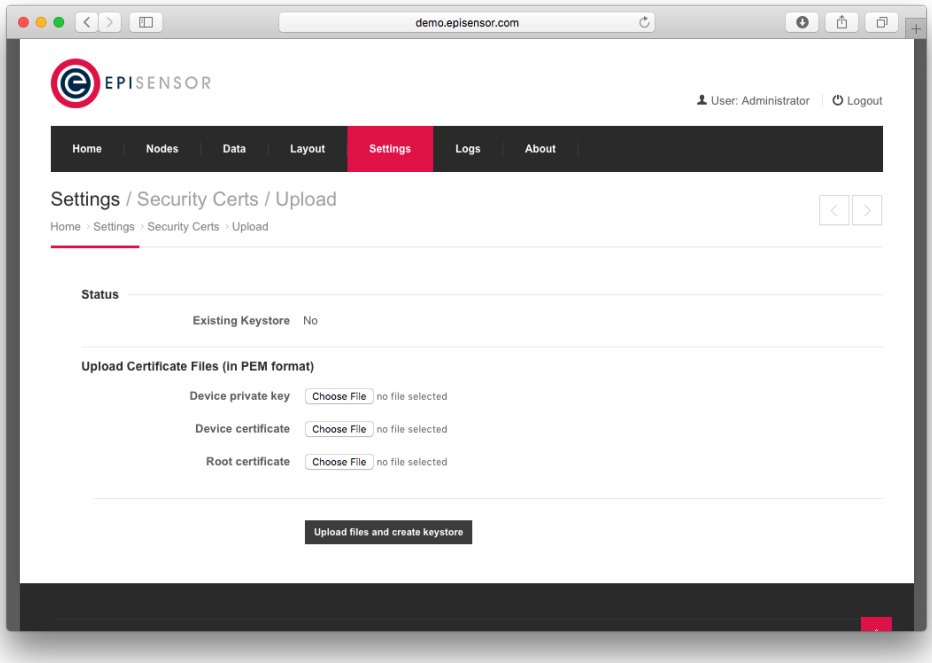
Make sure "Export Enabled" is checked, set the 'Reporting Mode' to "Snap to Clock", then click 'Save Changes'. This will tell the Gateway that data from this sensor should be sent to AWS IoT, as opposed to just being shown on the Gateway's web interface.



## *Uploading the security certificates*

We are now ready to upload the AWS IoT security certificates to the Gateway. The security certificates must be in place on the EpiSensor Gateway before any data can be exported by the Gateway to AWS IoT Core.

Security certificates for AWS IoT Core can be uploaded to the Gateway using the Settings → Security Certs → Upload page as shown in the following screenshot:

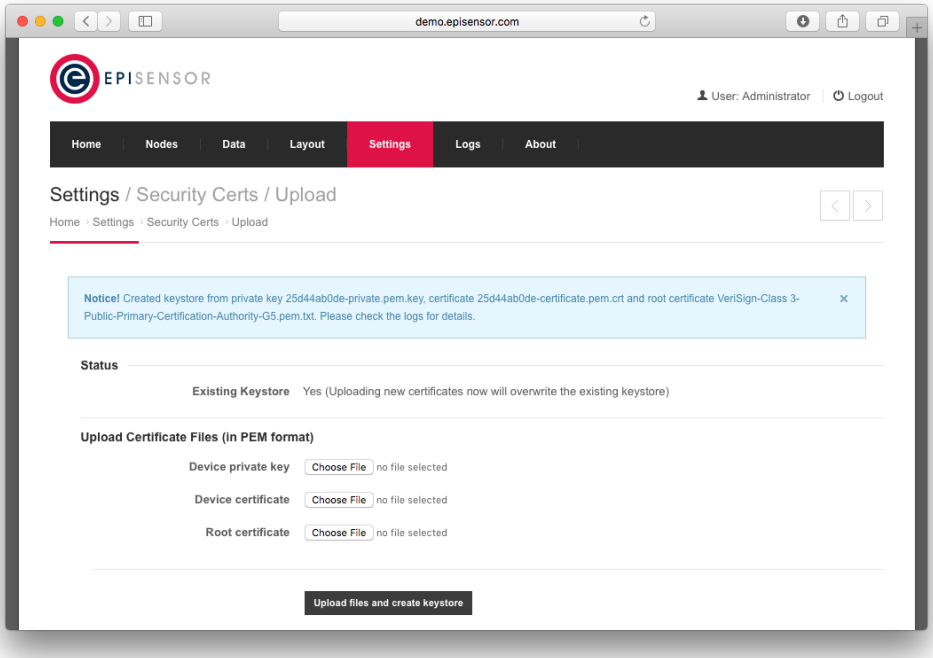



The EpiSensor Gateway expects the security certificate files to be in PEM format. The device private key file, the device certificate file and the root certificate file are all required. The device public key is not required. These files will be available in the correct format if the security certificates are generated within AWS IoT Core as described earlier.

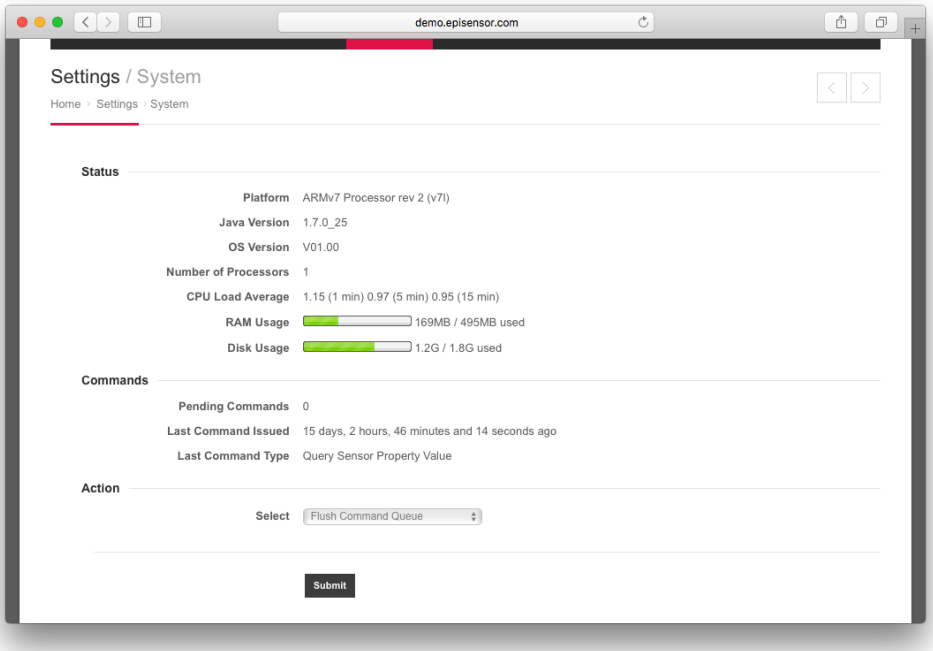
| File Name Example | Type | Upload Field Name |
|---|---|---|
| 22d44ab0de-certificate.pem.crt | Certificate | Device Certificate |
| 22d44ab0de-private.pem.key | Private Key | Device Private Key |
| 22d44ab0de-public.pem.key | Public Key | Not Required |
| VeriSign-Class3-Public-Primary-Certification-Authority-G5.pem.txt | Root Certificate | Root Certificate |

When the files are uploaded to the EpiSensor Gateway, a Java Keystore (JKS) file is generated from the uploaded certificates. The Status section of the Settings → Security Certs → Upload page will indicate whether or not there

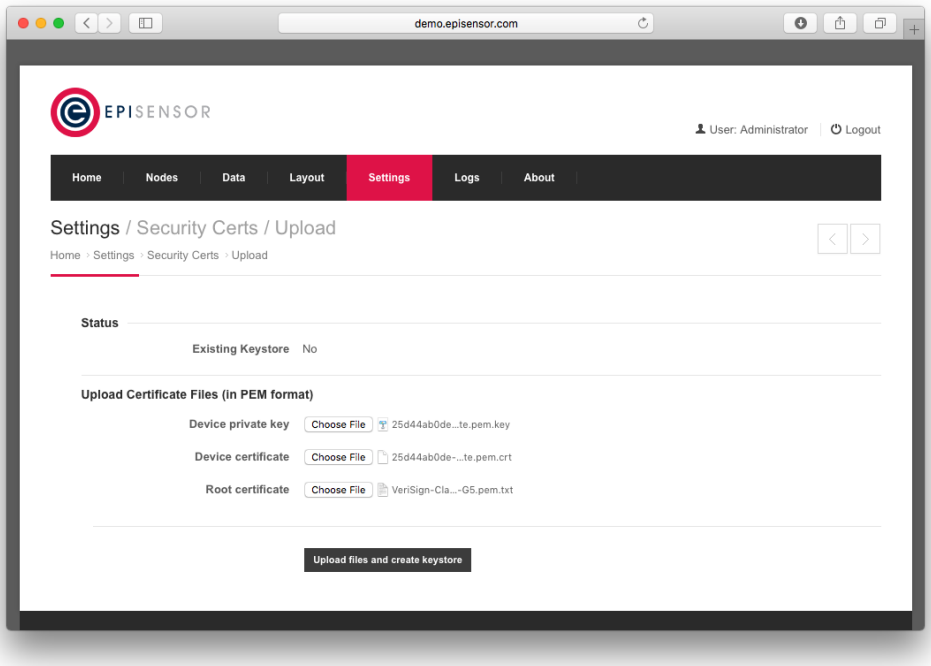
is an existing keystore file on the Gateway. If there is an existing keystore on the gateway, it will be overwritten by the newly created keystore.




In some scenarios, it might be required to delete the existing keystore on the Gateway. An example scenario would be when the Shared Access Signature (SAS) token is used instead of a certificate. The existing keystore may be deleted from the Settings → System page as shown in the following screenshot.

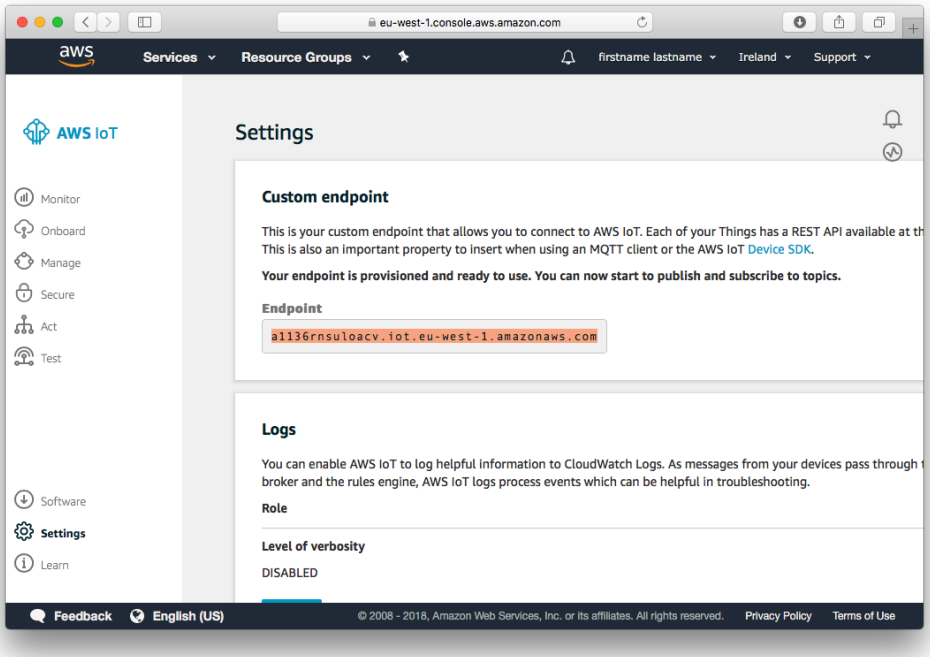


Once the keystore has been deleted, the Status on the Settings → Security Certs → Upload page should reflect that as shown in the following screenshot.




## *Enable MQTT Data Export*

This section describes the settings required on the Settings → Data Export page of the EpiSensor Gateway to connect to AWS IoT Core. To get the custom endpoint for your instance of AWS IoT Core, go to your AWS IoT console and click “Settings” on the bottom left menu, and you’ll see the custom endpoint as highlighted in the screenshot below. Copy this value for use in the steps below.




There are a number of different data export mechanisms available on the EpiSensor Gateway and each has different configuration options as shown on the Data Export page.


These are described in the following table:

| Data Export General Settings | Description |
|---|---|
| Data Export Type | This drop down list shows all the supported Data Export Formats and Transport Mechanisms. For exporting to AWS IoT, select “JSON via MQTT” on the data export page as shown in the following screenshot. |
| Data Export Interval | The ‘Data Export Interval’ field specifies the maximum time interval (in minutes) between attempted exports. The EpiSensor Gateway will export |

| | |
|---|---|
| | when either the 'Max Data Points per Export' has been reached or the 'Data Export Interval' has elapsed. |
| Max Data Points per Export | The 'Max Data Points per Export' field specifies the maximum number of discrete data points that will be accumulated on the Gateway before exporting. The EpiSensor Gateway will export when either the 'Max Data Points per Export' has been reached or the 'Data Export Interval' has elapsed. |
| Live Stream | Live Stream export means data points will be exported as soon as possible. If 'Live Stream' export is selected, both the 'Data Export Interval' and 'Max Data points per Export' settings are overridden. In other words the EpiSensor Gateway will export data as soon as it has arrived and been processed by the EpiSensor Gateway. This can be useful when data is required to be as real-time as is possible. However, it can also result in inefficient use of the data transport mechanism because the Gateway will be exporting smaller amounts of data more frequently. |

In addition, there are configuration options specifically required for MQTT via JSON export type as described in the following table:

| **MQTT via JSON Settings** | **Description** |
|---|---|
| MQTT Broker URL | This should be configured with the endpoint of your AWS IoT Core. Paste the value you copied above into this field. |
| Encryption | Encryption may be enabled (On) or disabled (Off). If enabled, SSL will be used to make an encrypted connection to AWS IoT Core. For this configuration, port 8883 should be open in any firewall between the EpiSensor Gateway and the Internet. If disabled, TCP the connection will not be encrypted and TCP port 1883 will be used. |
| QOS Level | Quality of Service Level for published data. Level 0, 1 or 2 may be selected. AWS IoT Core supports QoS values of 0 or 1. |
| Support AWS Device Shadow | Support for the AWS IoT Device Shadows may be enabled or disabled. |
| Client ID | The unique client ID for this client (publisher). If this field is left empty the EpiSensor Gateway will use its Serial Number as the client ID in the connection to AWS IoT Core. |
| Username | The username when client authentication is to be used. This field is not required for connection to AWS IoT Core. |

| | |
|---|---|
| Password | The password when client authentication is to be used. This field is not required for connection to AWS IoT Core. |
| MQTT Data Publish Topic | The topic that the EpiSensor Gateway will publish data to. In the screenshot above, the topic is set to “test_gateway/data” but this can be set to any value. |

If for any reason the export of data fails (for example lost internet connection or expired security certificates), data which failed to export will be saved on the Gateway. The export of this data will be retried based on the number of minutes defined in the ‘Data Export Interval’ field.

Files queued for export can be deleted from the system using the drop down list on the Settings → System page.

## Testing and Next Steps

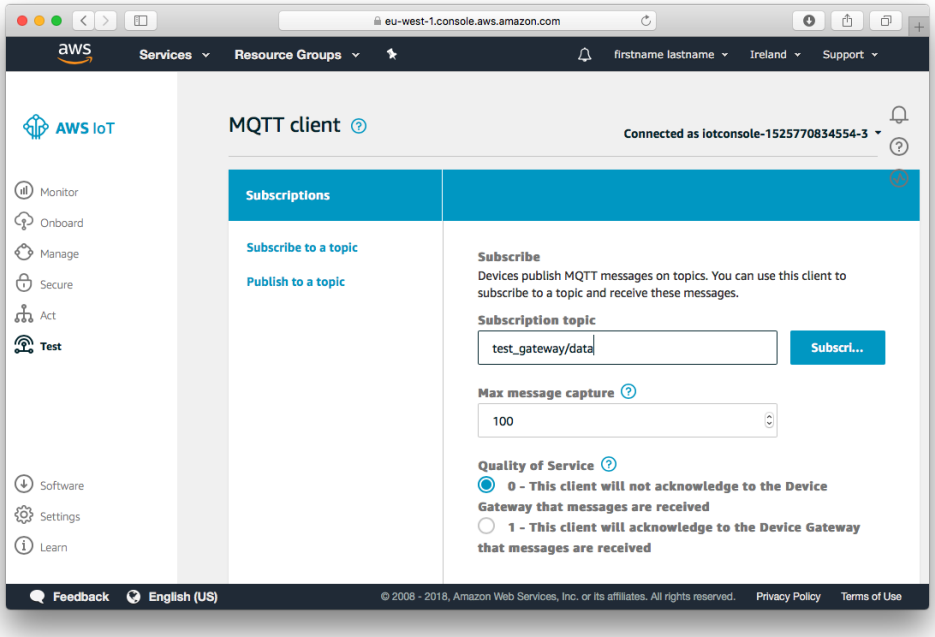
The “Last Data Export” field on the Settings → Data Export page of the EpiSensor Gateway will show how long ago data was last sent to the server.

If you refresh the page and this field shows a recent data export (i.e. less than the data export interval you defined) then the connection to AWS IoT Core has been established!

Depending on how the reporting intervals of the sensors are configured and the data export interval, there may be a delay in data being sent to AWS IoT Core.

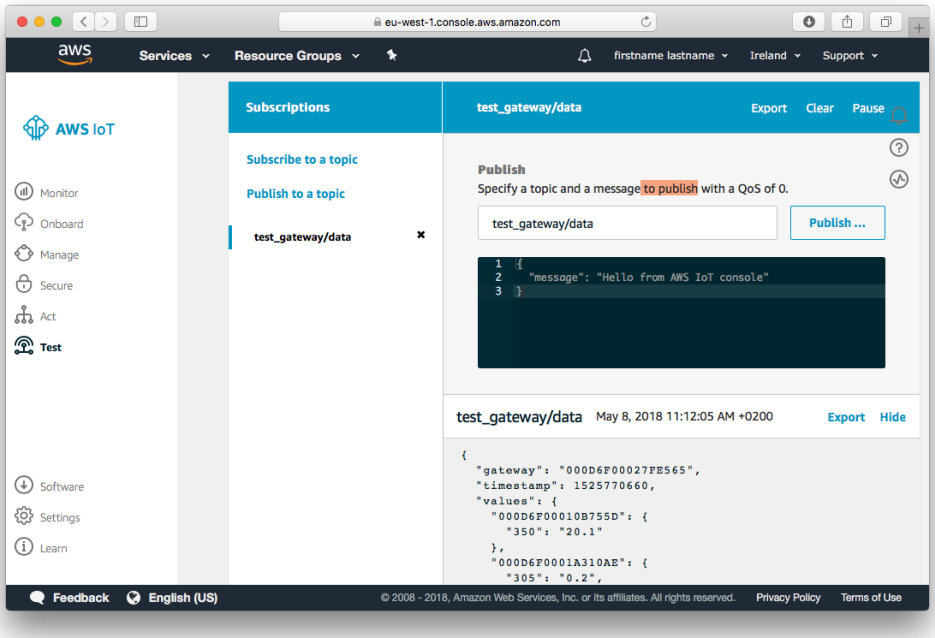
### *Testing the Connection from AWS IoT Core*

You can also view live data within the AWS IoT console by clicking on ‘Test’ from the menu on the left of the AWS IoT Console.




On this screen, click “Subscribe to a topic”, and set the topic to the Serial Number of the Gateway you have been working with (unless of course you defined a different topic to export to on the Settings → Data Export page).

A short time later, you should see sensor data arriving in JSON format, as shown on the screenshot below:



### *Working with the sensor data*

Data from sensors of the nodes joined to the EpiSensor Gateway is published to AWS IoT Core in the following format:


```
{
  "gateway": "000D6F00027FE565",
  "timestamp": 1525770720,
  "values": {
    "000D6F00010B755D": {
      "350": "20.1"
    },
    "000D6F0001A310AE": {
      "305": "0.2",
      "307": "237.9",
      "308": "237.2",
      "309": "177.86",
      "324": "145.0",
      "328": "150.0",
      "335": "0.266"
    },
    "000D6F0001A31E69": {
      "505": "230.0"
    }
  }
}
```


In the example above, the Serial Number of the EpiSensor Gateway publishing the data is 000D6F00027FE565.

The timestamp of the received data is 1525770720 (seconds since the Epoch). The data is then group per node connected to the EpiSensor Gateway. In this example, we have one data point from a node with serial number “000D6F00010B755D” at that timestamp from a sensor with ID = 350, and the value of the data point is 20.1.

## Device Shadows

AWS IoT device shadow service maintains a ‘shadow’ for each device connected to AWS IoT. The shadow may be used to get and set the state of a device over MQTT. Each device's shadow is uniquely identified by the name of the corresponding ‘Thing’.

### *Enable Device Shadow Support*

Device Shadow support is enabled on the EpiSensor Gateway using the ‘Support AWS Device Shadow’ radio button on the Settings → Data Export page. When AWS Device Shadow support is enabled on the Gateway, it will publish updates to the device shadow’s topic, which will have the following structure:

```
$aws/things/THING_ID/shadow/update
```

In addition the Gateway will subscribe to the following topic(s) and react to requests to update the device shadow:

```
$aws/things/THING_ID/shadow/update/delta
```

## *What do the Device Shadow updates look like?*

When a data point is received by the Gateway, a message to update the Device Shadow will be published to the corresponding topic for that node, as per the following example:

**$aws/things/000D6F0001A30F78/shadow/update**


```
{
  "previous": {
    "state": {
      "reported": {
        "350": "57.8"
      }
    },
    "metadata": {
      "reported": {
        "350": {
          "timestamp": 1525787344
        }
      }
    },
    "version": 56
  },
  "current": {
    "state": {
      "reported": {
        "350": "58.0"
      }
    },
    "metadata": {
      "reported": {
        "350": {
          "timestamp": 1525787434
        }
      }
    },
    "version": 57
  },
  "timestamp": 1525787434
}
```


In the above example “350” is the ID of the sensor on node 000D6F0001A30F78. The data value is “58.0” and the data timestamp is 1525787434 (seconds since the Epoch).

### *What will trigger a device shadow update?*

A device shadow update will be triggered when the EpiSensor Gateway receives a data point from a sensor of a node for which export is enabled to AWS IoT. The device shadow update occurs for the discrete data point of that sensor and happens independently of the bulk publish of data as configured on the Settings → Data Export page.

## Ordering Information

EpiSensor products are available to order directly or via EpiSensor's distribution partners. The following table lists the available Gateways and Starter Kits that are compatible with AWS IoT Core.

| SKU | Description |
|---|---|
| NGR-30-3 | Ethernet communications, incl. 1yr EpiSensor Gateway software license, up to 50 nodes/200 sensors |
| NGR-30-5 | Dell Edge Gateway 3002, incl. 1yr EpiSensor Gateway software license, up to 100 nodes/1000 sensors |
| ASAVIE-SKV1 | Industrial IoT Accelerator Kit (in partnership with Asavie and Dell), incl. Dell Edge Gateway 3002 with EpiSensor Gateway software, 2x wireless temperature sensors |

## Troubleshooting & Support

If you are experiencing problems with your NGR Gateway or any other part of your EpiSensor system, or you notice something unusual - please contact EpiSensor support at the following email address, phone number or via live chat on our website.

- Email: support@episensor.com
- Tel: +353 61 512 500
- Website: http://episensor.com

For customers and partners who are deploying systems in business-critical environments, there are a number of support packages available that offer a higher level of service and response time. For more information on EpiSensor Premium Support, visit: http://episensor.com/premium-support/

## Warranty

All EpiSensor products and provided with a 365 day limited warranty effective from the shipping/invoice date of an order. During the warranty period, under the conditions of normal use, EpiSensor will repair or replace any product that has a manufacturing defect.

Warranty can be extended by up to 4 years within 30 days of a purchase. For more information on warranty, visit: http://episensor.com/warranty/

## Glossary

Definitions for terms and abbreviations used in this document are listed in the following table:

| Term | Description |
|---|---|
| Allow join mode | A mode that can be enabled on the Gateway that allows new wireless nodes to join |
| AWS | Amazon Web Services |
| Gateway | The central computer that managed the EpiSensor system |
| Interval and Delta | Reporting mode where data is produced when the reporting interval has elapsed, unless a change is detected |
| JKS | Java Keystore |
| MQTT | MQTT (Message Queuing Telemetry Transport) is an ISO standard publish-subscribe-based messaging protocol |
| Node | Used to describe a physical EpiSensor product |
| Reporting Interval | The length of time between each data point produced by a node |
| Reporting Mode | Defines how an EpiSensor node should report data to the Gateway |
| SAS | Shared Access Signature |
| Sensor | Describes a feed of data within the EpiSensor system |
| Snap to Clock | Reporting mode where data is 'snapped' to the nearest 1 minute / 5 minute / 15 minute interval etc. |
| WSN | Wireless Sensor Network |
| ZigBee | IEEE 802.15.4 Wireless communications standard that EpiSensor nodes use. |