

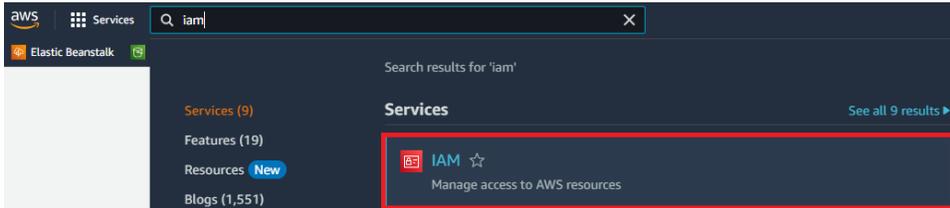
# Step 1 Set up AWS IoT

Before using the btibAWS IoT you must first have an AWS account, follow this link to do so: <https://aws.amazon.com/premiumsupport/knowledge-center/create-and-activate-aws-account/>

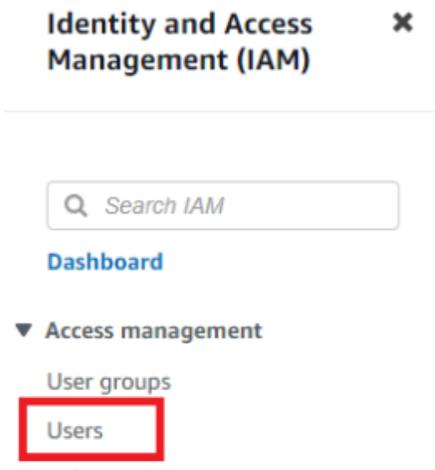
## Setup API Key

Niagara needs an **API key** to access AWS IoT Services and manage devices:

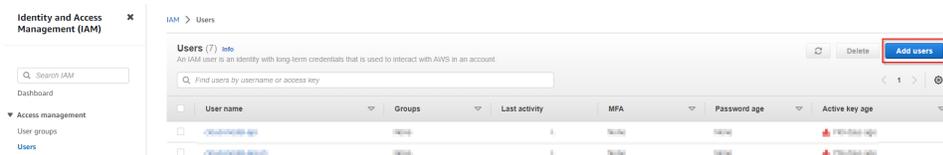
1. Go to the **IAM** service on the **AWS console**.



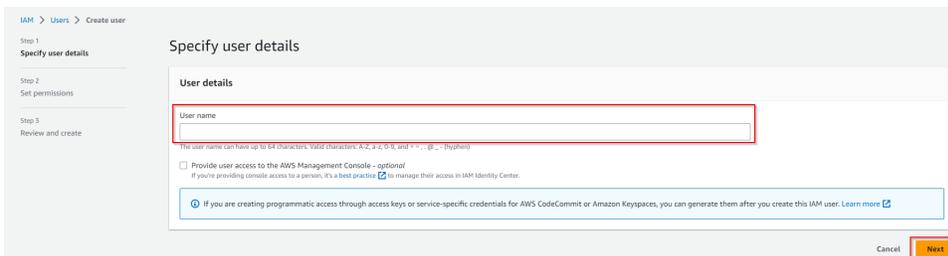
2. Then **Users**



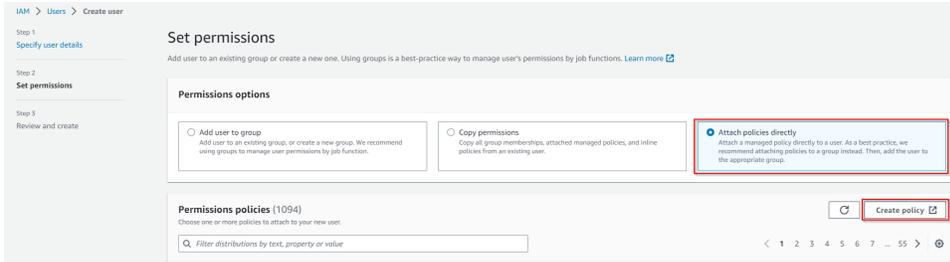
3. Click the **Add Users** button



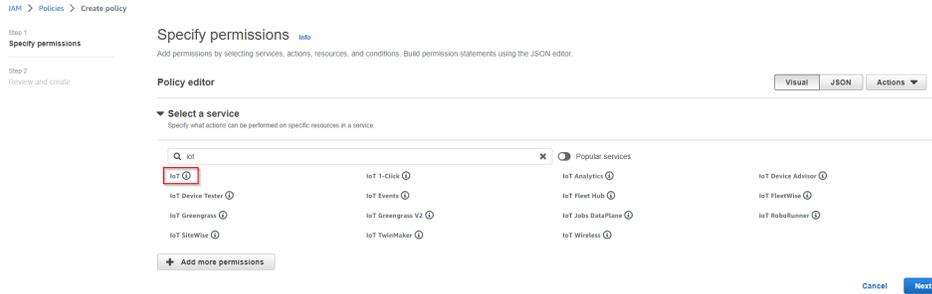
4. Give a **username** to your user, then click next.



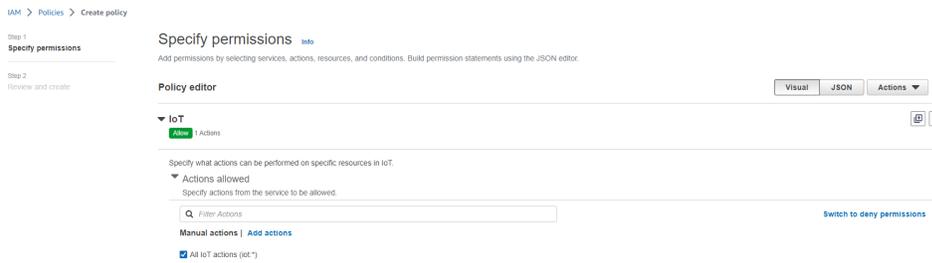
5. Then **Attach policies directly**, Then hit **Create policy**. A new tab will open.



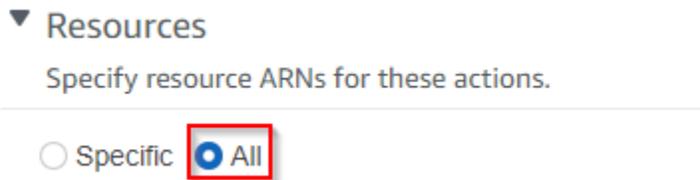
6. Select the **IoT** service



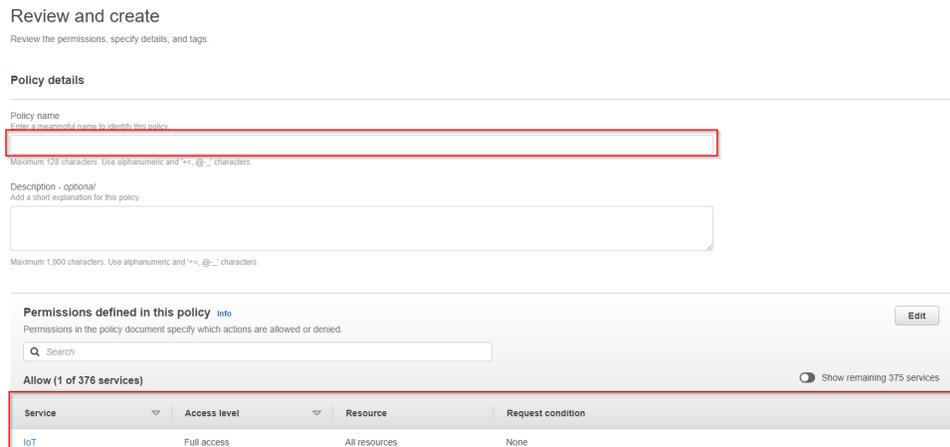
7. Allow **All IoT** actions



8. Allow **All** resources, then hit **Next**



9. Give your policy a **name**, make sure that you have **full access** on the summary. finally hit **Create policy**



10. Now go back to the "Add user" page hit **refresh** (top right), look for your policy on the search field, select it and click **Next**.

**Set permissions**  
Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

**Permissions options**

- Add user to group  
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.
- Copy permissions  
Copy all group memberships, attached managed policies, and inline policies from an existing user.
- Attach policies directly  
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

**Permissions policies (1094)**  
Choose one or more policies to attach to your new user.

Refresh Create policy

Filter distributions by text, property or value 1 match

niagara X Clear filters

<input type="checkbox"/>	Policy name	Type	Attached entities
<input checked="" type="checkbox"/>	niagara-aws-iot-full-access	Customer managed	

11. Add tags (optional). Then hit **Next**

12. Finally hit **Create User**.

13. Your User was successfully created. Click on **View user**

✔ **User created successfully**  
You can view and download the user's password and email instructions for signing in to the AWS Management Console. [View user](#)

14. Go in the **Security credentials** Tab and create an access key

IAM > Users > niagara-aws-iot-demo

niagara-aws-iot-demo [Delete](#)

**Summary**

ARN arn:aws:iam::178126363112:user:niagara-aws-iot-demo	Console access Disabled	Access key 1 Not enabled
Created May 11, 2023, 17:59 (UTC-02:00)	Last console sign-in -	Access key 2 Not enabled

Permissions | Groups | Tags | **Security credentials** | Access Advisor

**Console sign-in** [Enable console access](#)

Console sign-in link  
<https://178126363112.signin.aws.amazon.com/console>

Console password  
Not enabled

**Multi-factor authentication (MFA)** (0)  
Use MFA to increase the security of your AWS environment. Signing in with MFA requires an authentication code from an MFA device. Each user can have a maximum of 8 MFA devices assigned. [Learn more](#)

[Remove](#) [Resync](#) [Assign MFA device](#)

Device type	Identifier	Created on
No MFA devices. Assign an MFA device to improve the security of your AWS environment.		

[Assign MFA device](#)

**Access keys (0)** [Create access key](#)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. [Learn more](#)

15. Select **Third-party service**, check the "I understand..." checkbox and click **Next**

**Access key best practices & alternatives**  
Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

- Command Line Interface (CLI)  
You plan to use this access key to enable the AWS CLI to access your AWS account.
- Local code  
You plan to use this access key to enable application code in a local development environment to access your AWS account.
- Application running on an AWS compute service  
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.
- Third-party service**  
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.
- Application running outside AWS  
You plan to use this access key to enable an application running on an on-premises host, or to use a local AWS client or third-party AWS plugin.
- Other  
Your use case is not listed here.

**Alternative recommended**  
As a best practice, use temporary security credentials (IAM roles) instead of creating long-term credentials like access keys, and don't create AWS account root user access keys. [Learn more](#)

I understand the above recommendation and want to proceed to create an access key.

[Cancel](#) [Next](#)

16. Click on **Create access key**

## Set description tag - optional

The description for this access key will be attached to this user as a tag and shown alongside the access key.

Description tag value  
Describe the purpose of this access key and where it will be used. A good description will help you rotate this access key confidentially later.

Maximum 256 characters. Allowed characters are letters, numbers, spaces representable in UTF-8, and: \_ : / = + - @

Cancel Previous **Create access key**

17. Retrieve you **Access keys** (either copy paste your values or download the .csv file). Keep them they will be needed to setup the connector in your workbench

## Retrieve access keys

**Access key**  
If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key	Secret access key
AKIASS6JITHUORSIM4FR	***** Show

**Access key best practices**

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the [Best practices for managing AWS access keys](#).

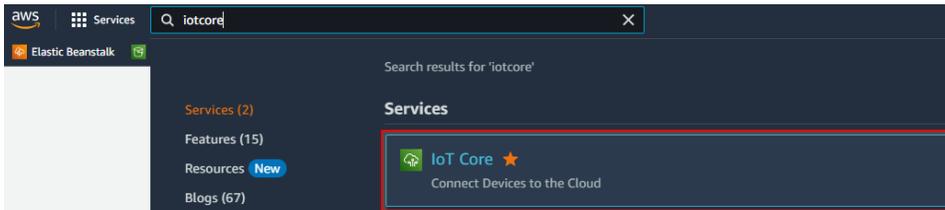
Download .csv file Done

## Setup Devices certificates

AWS uses Asymmetric keys for device authentication and authorization.

To create a key pair and a certificate follow these steps:

1. Go to the **IoT Core** service on the **AWS console**.



2. Then security Certificates

## Monitor

### Connect

- Connect one device
- ▶ Connect many devices

### Test

- ▶ Device Advisor
- MQTT test client
- Device Location [New](#)

### Manage

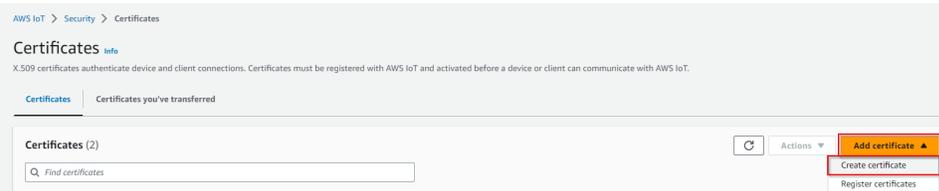
- ▶ All devices
- ▶ Greengrass devices
- ▶ LPWAN devices
- ▶ Remote actions
- ▶ Message routing
- Retained messages

### ▼ Security

#### Intro

#### Certificates

3. On the top right corner hit **Add certificate**.



4. Then Select **Auto-generate new certificate**, select **Active** and hit **Create**

## Create certificate [Info](#)

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.

Cancel

Create

5. Download the certificate, the public key (optional) and the private key

## 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  **Download**  
 70a538e6dd3...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  **Download**  
 70a538e6dd35d6fa08922e0...957c485-public.pem.key

Private key file  **Download**  
 70a538e6dd35d6fa08922e0...57c485-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  **Download**  
 RSA 2048 bit key: Amazon Root CA 1

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

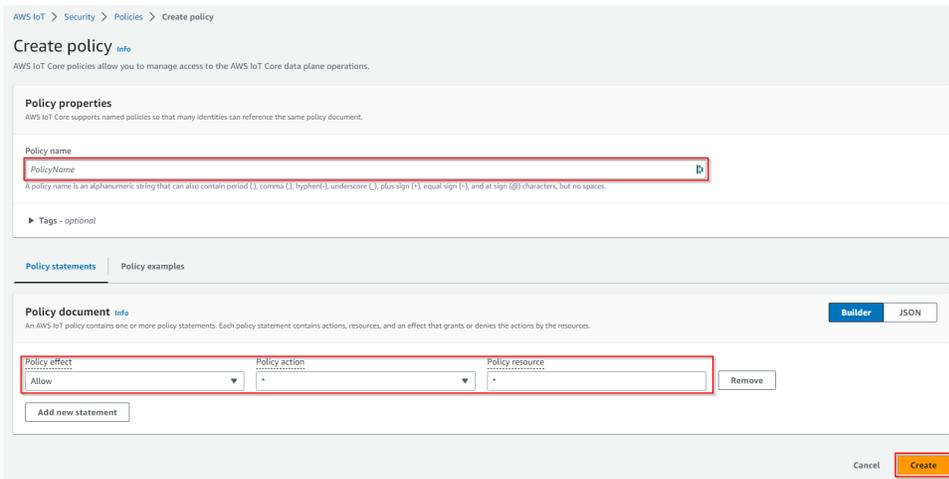
If you don't see the root CA certificate that you need here, AWS IoT supports additional root CA certificates. These root CA certificates and others are available from our developer guides.

**Continue**

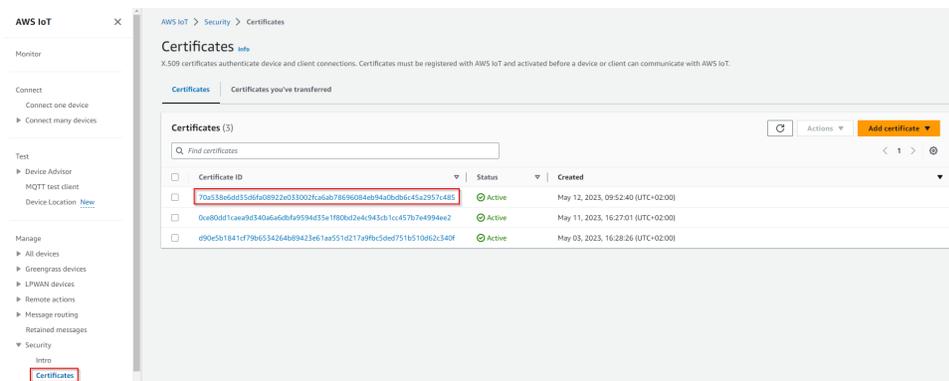
6. You will also need the **AWS CA key file**, you can download it here: [VeriSign-Class 3-Public-Primary-Certification-Authority-G5.pem](#).
7. Now go to **Security > Policies** and hit **Create Policy**

The screenshot shows the AWS IoT console interface. On the left-hand side, there is a navigation menu with 'Security' and 'Policies' highlighted with red boxes. The main content area is titled 'AWS IoT policies (1) info' and includes a search bar, a 'Delete' button, and a 'Create policy' button. Below the search bar, there is a table with one policy listed: 'niagara\_test\_full'.

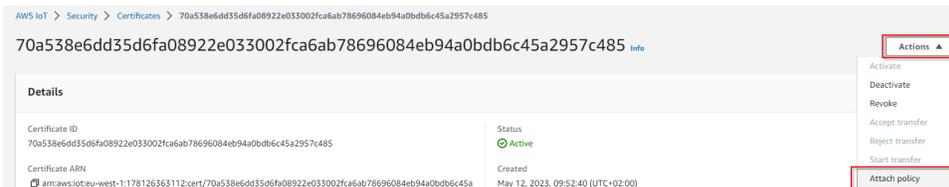
8. Give your policy a **name**. select the **"Allow"** policy effect, and put **"\*\*"** in the policy action and policy resource. Then hit **Create**



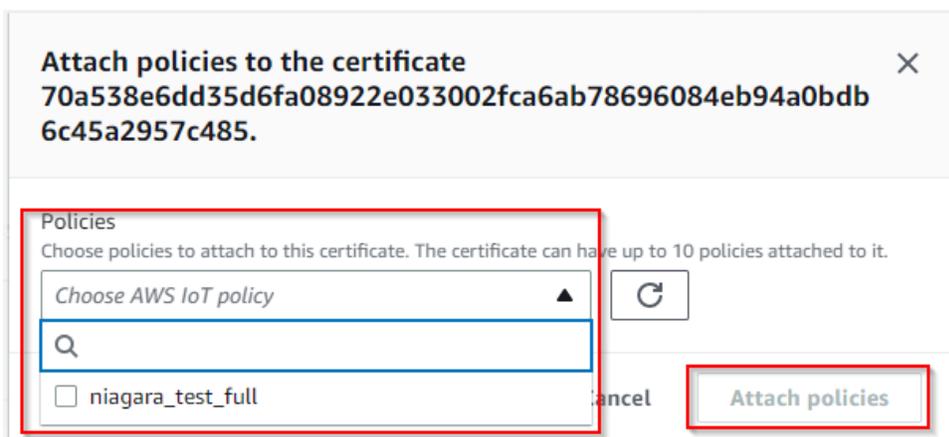
9. Go back to certificates. Choose the certificate you created earlier (check the date).



10. Under Actions select **Attach policy**



11. Select your policy then hit **Attach**.



12. Now note down your **certificate ARN**, we will need it later.

70a538e6dd35d6fa08922e033002fca6ab78696084eb94a0bdb6c45a2957c485 [Info](#)

**Details**

Certificate ID 70a538e6dd35d6fa08922e033002fca6ab78696084eb94a0bdb6c45a2957c485	Status ✔ Active
Certificate ARN <a href="#">arn:aws:iot:us-east-1:123456789012:certificate/70a538e6dd35d6fa08922e033002fca6ab78696084eb94a0bdb6c45a2957c485</a>	Created May 12, 2023, 09:52:40 (UTC+02:00)
	Valid

## API endpoint

Finally you will need your **API endpoint**

To find it follow these steps:

1. Go to the **IoT Core** service on the **AWS console**.

Search results for 'iotcore'

Services (2)  
Features (15)  
Resources **New**  
Blogs (67)

**IoT Core** ★  
Connect Devices to the Cloud

2. Go to **Settings**, and copy paste your endpoint

**AWS IoT** Settings

**Device data endpoint** [Info](#)

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  
[arn:aws:iot:us-east-1:123456789012:device-data-endpoint/70a538e6dd35d6fa08922e033002fca6ab78696084eb94a0bdb6c45a2957c485](#)

Select security policy [Info](#)  
To customize your TLS settings, such as TLS versions and supported cipher suites, choose a security policy.  
IoTSecurityPolicy\_TLS12\_1\_0\_2015\_01

[Compare security policies](#)

**Domain configurations** [Info](#)

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.

[Actions](#) [Create domain configuration](#)

Name	Domain name	Status	Service type	Date updated
No domain configurations You don't have any domain configurations.				

[Create domain configuration](#)

## Recap

Let's recap, after all these steps you should have 6 things:

- The credentials csv file for AWS user that contains the client access id and secret.
- The certificate file.
- The private key file.

- The public key file (optional).
- The AWS CA key file.
- The ARN certificate
- And last but not least the API Endpoint

Congrats !!! You finished the AWS setup go to next step:

## Next Step

---

[Step 2 Set up AWS connector for devices points and references](#)