Application Manual For Module Access to AWS IoT Platform

— 潘先强(Herbert Pan) —

CONTENTS

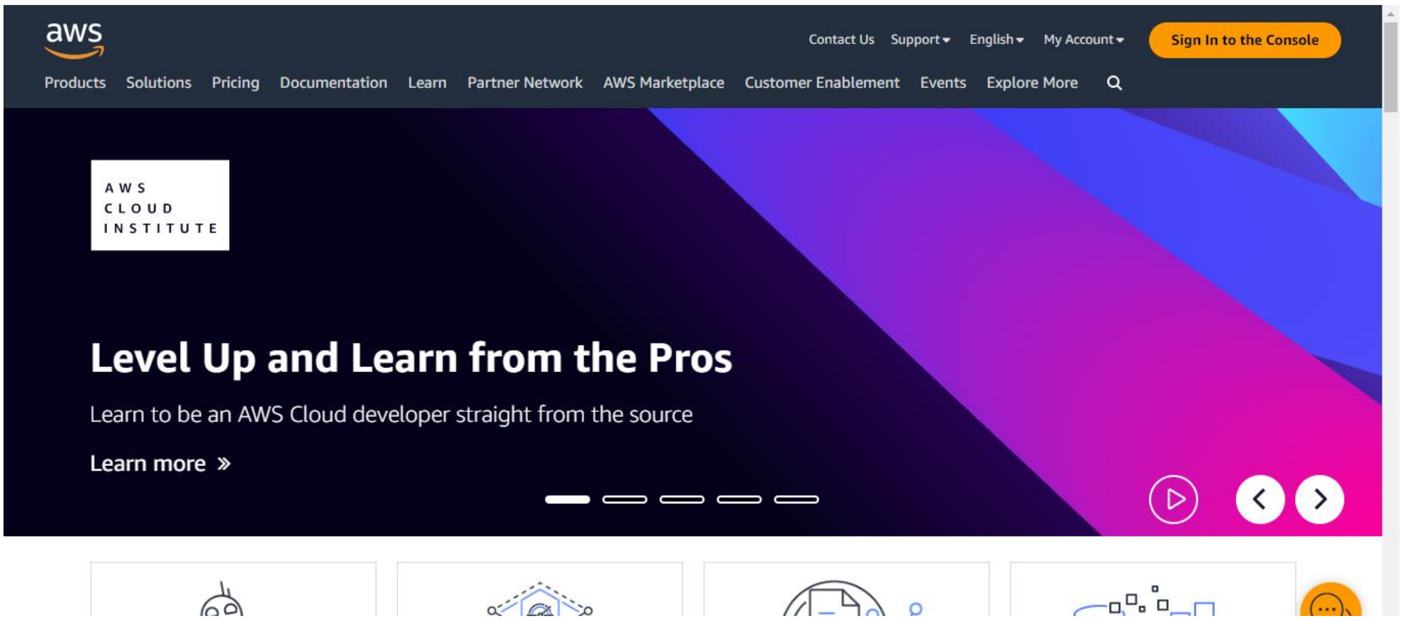
一、Forward

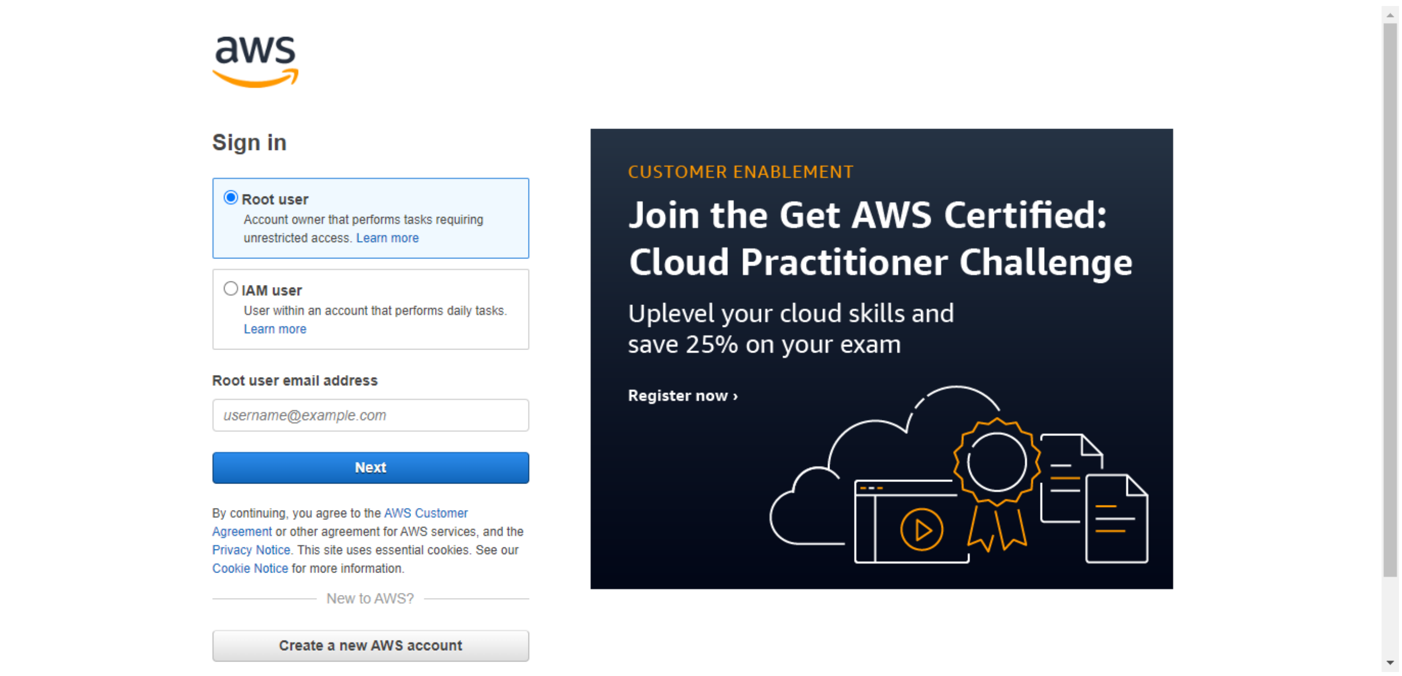
Currently, some customers will connect to Amazon AWS platform through MQTT protocol based on Quectel module; This document is aimed at the Quectel’s module access to the AWS platform and the use of MQTT application so that customers or other-relevant can understand docking methods and processes on the platform and module in a quick and high-effective way.

二、Login to AWS

AWS Platform：https://aws.amazon.com/?nc1=h_ls

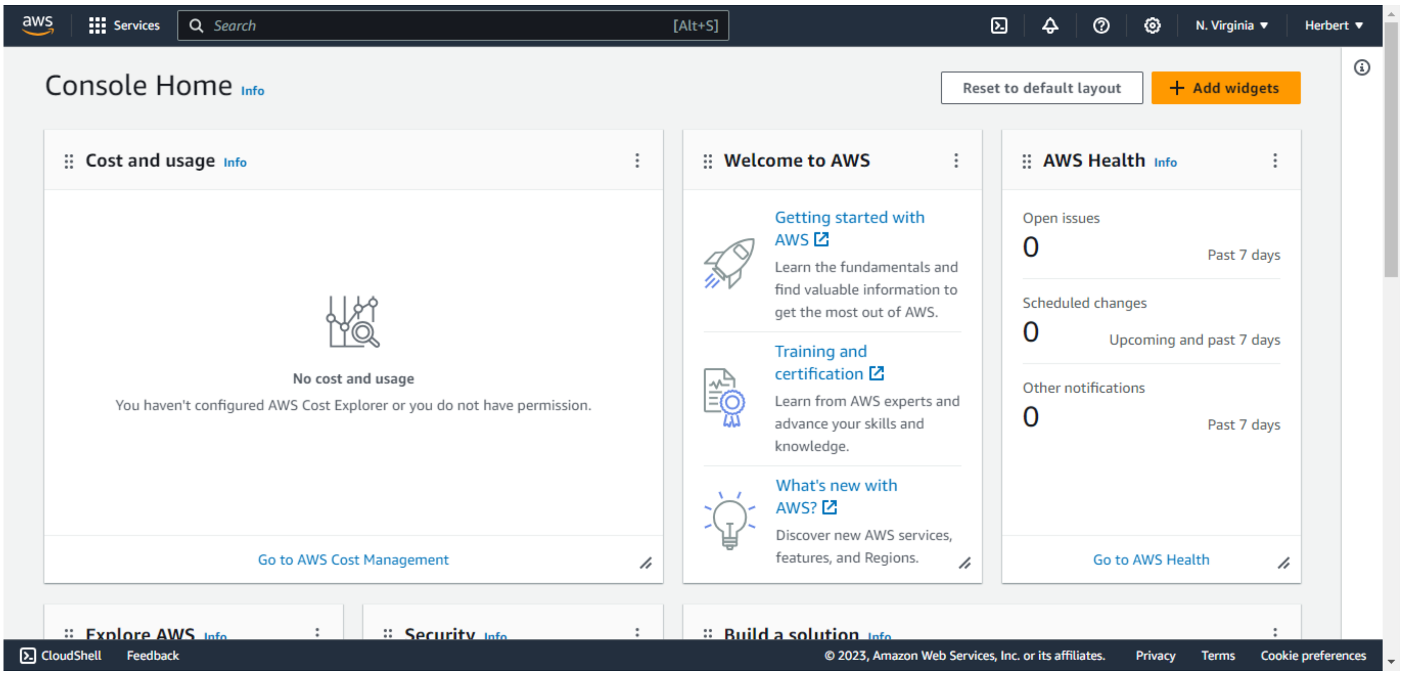
By the applied login account, click "Sigin In to the Console" to login and enter the home page of the console, as shown below;



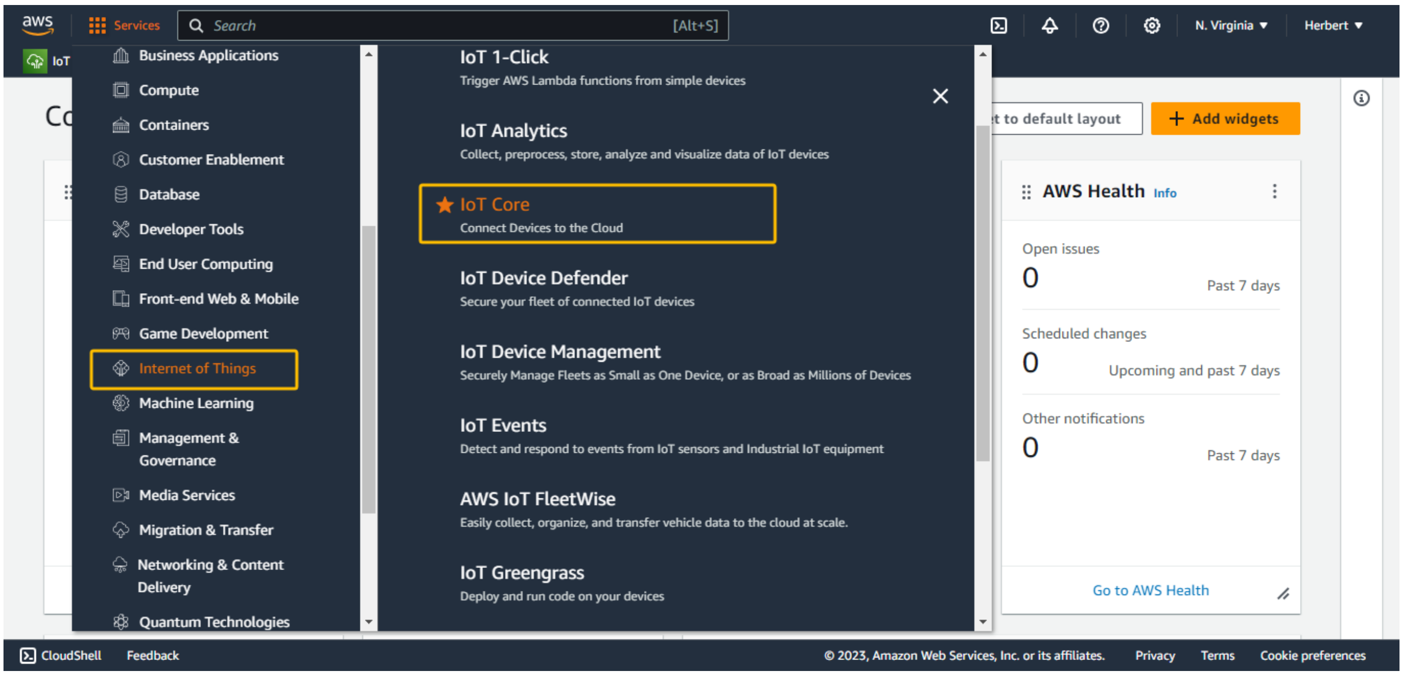

aws
Sign in
Root user
Account owner that performs tasks requiring unrestricted access. Learn more
IAM user
User within an account that performs daily tasks. Learn more
Root user email address
username@example.com
Next
New to AWS?
Create a new AWS account
CUSTOMER ENABLEMENT
Join the Get AWS Certified: Cloud Practitioner Challenge
Uplevel your cloud skills and save 25% on your exam
Register now ›

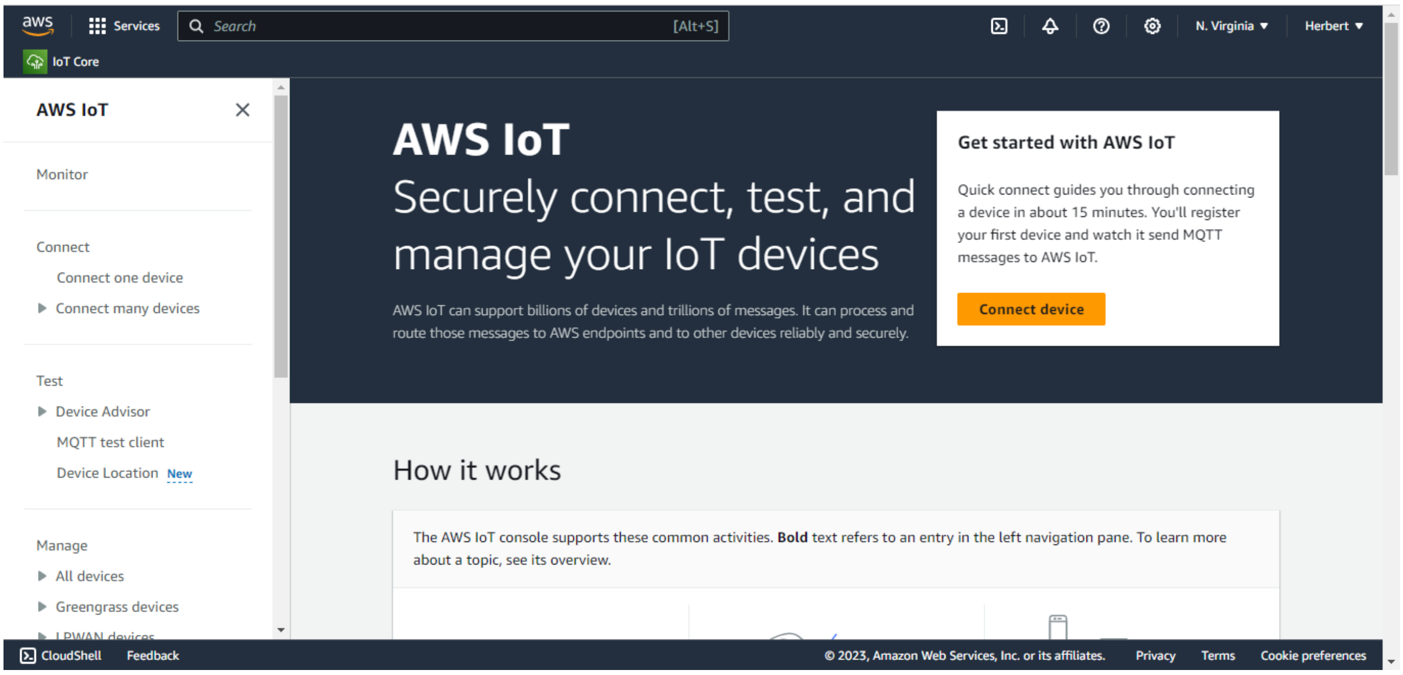
By continuing, you agree to the AWS Customer Agreement or other agreement for AWS services, and the Privacy Notice. This site uses essential cookies. See our Cookie Notice for more information.

AWS control console：https://us-east-1.console.aws.amazon.com/console/home?region=us-east-1



Click "Services → Internet of Things → IoT Core" to enter the home page of AWS IoT,as shown below.





AWS IoT
Securely connect, test, and manage your IoT devices

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

Get started with AWS IoT

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

Connect device

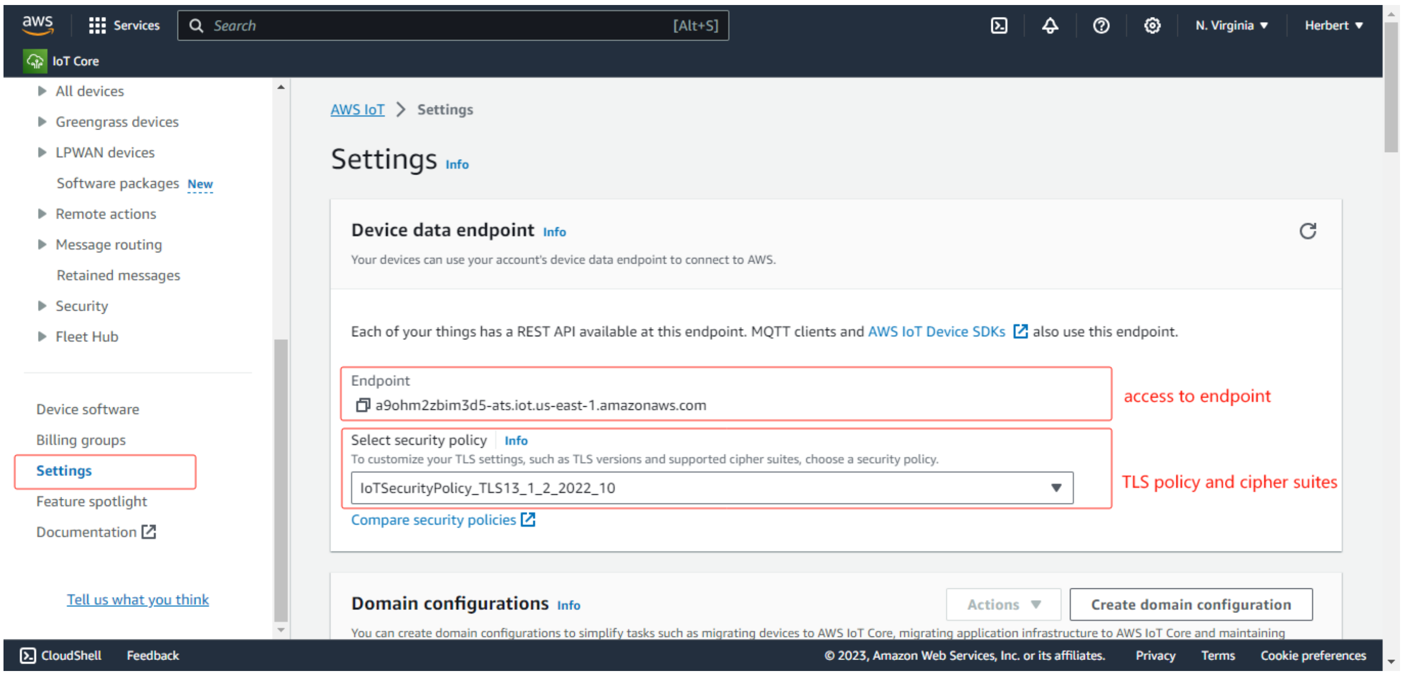
How it works

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

三、 AWS IoT Platform Settings

3.1 Device Data Endpoint

As shown below, you can access the Settings page through "Settings" in the control panel on the left side of the AWS IoT page, view the device data endpoint to be accessed and select a security policy (note that the security policy is associated with the cipher suites,so you need to select the cipher suite and security policy supported by the module).



Settings Info

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
a9ohm2zbim3d5-ats.iot.us-east-1.amazonaws.com

Select security policy Info
To customize your TLS settings, such as TLS versions and supported cipher suites, choose a security policy.
IoTSecurityPolicy_TLS13_1_2_2022_10

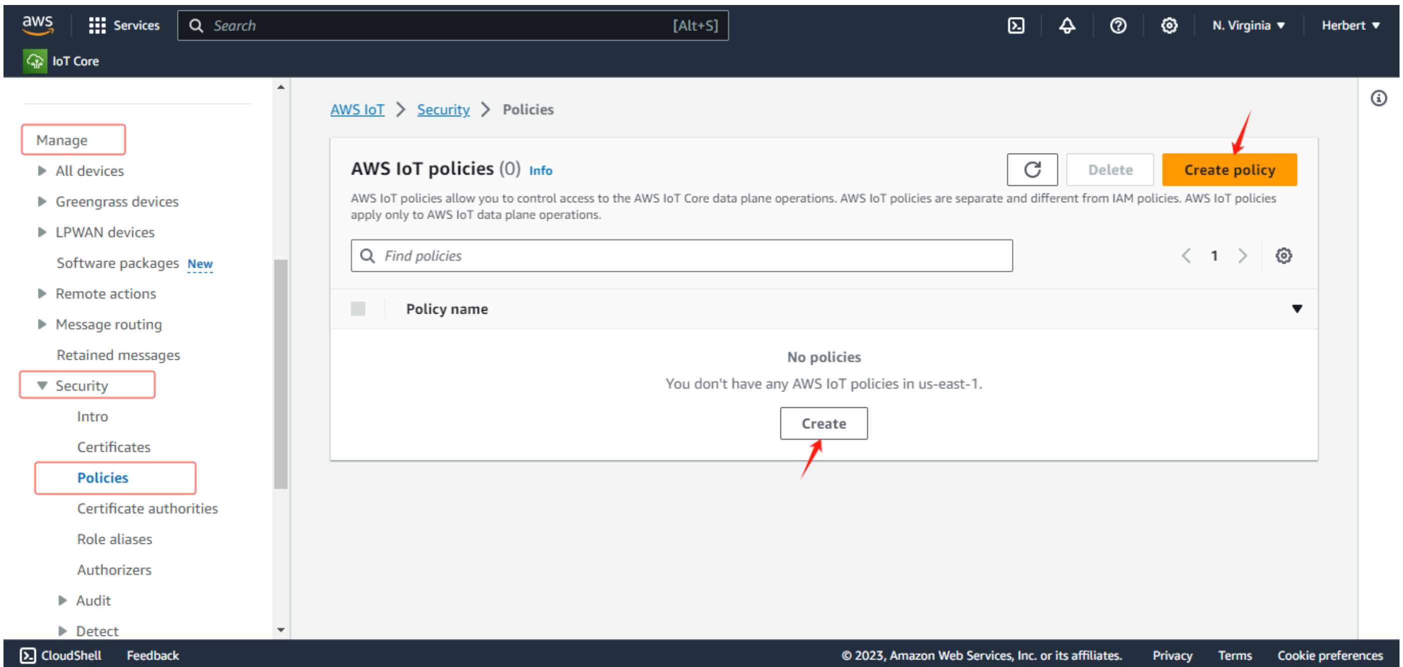
Compare security policies

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

3.2 AWS IoT Policies

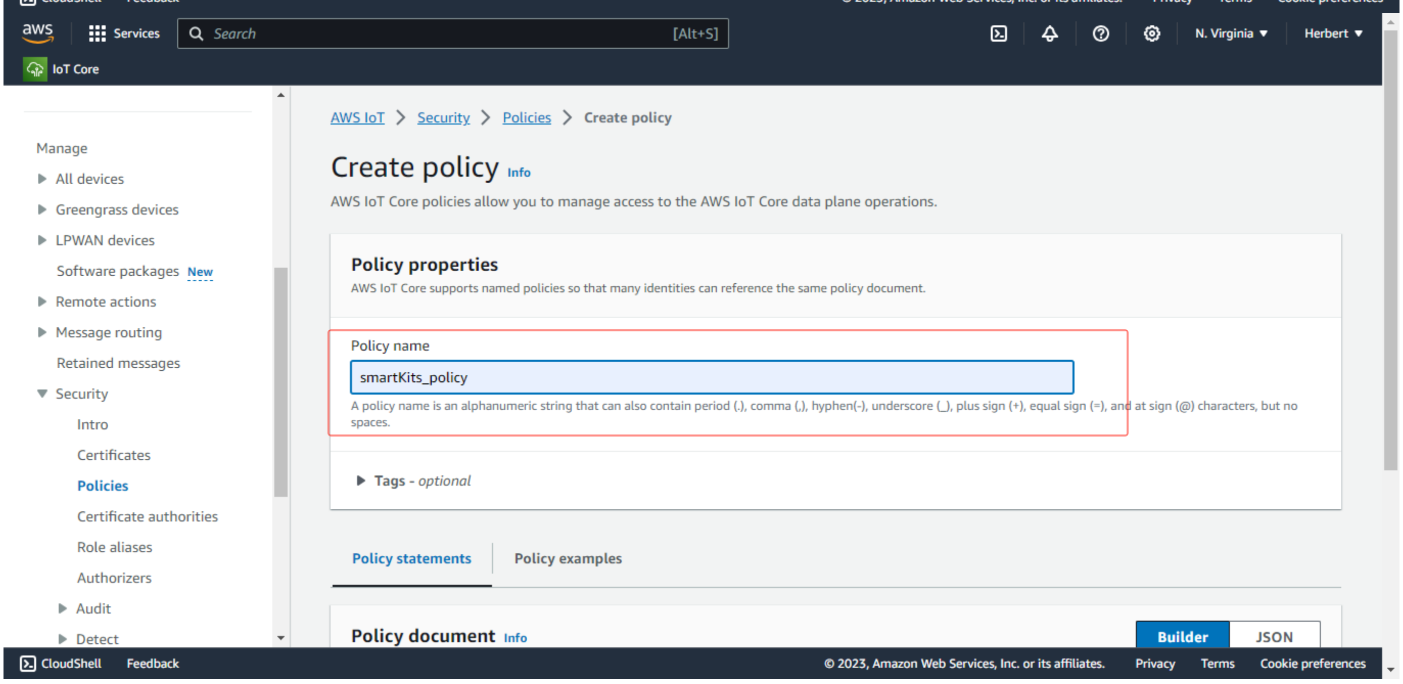
As shown below, to enter the policy configuration page through "Manage → Security → Policies" in the control panel on the left side of the AWS IoT page, click "Create Policy", and configure related policy attributes. In the policy statement, set Policy Effect to Allow, Policy Action to * (all AWS IoT operations), and Policy Resource to *.


AWS IoT > Security > Policies

AWS IoT policies (0) Info

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

No policies
You don't have any AWS IoT policies in us-east-1.


AWS IoT > Security > Policies > Create policy

Create policy Info

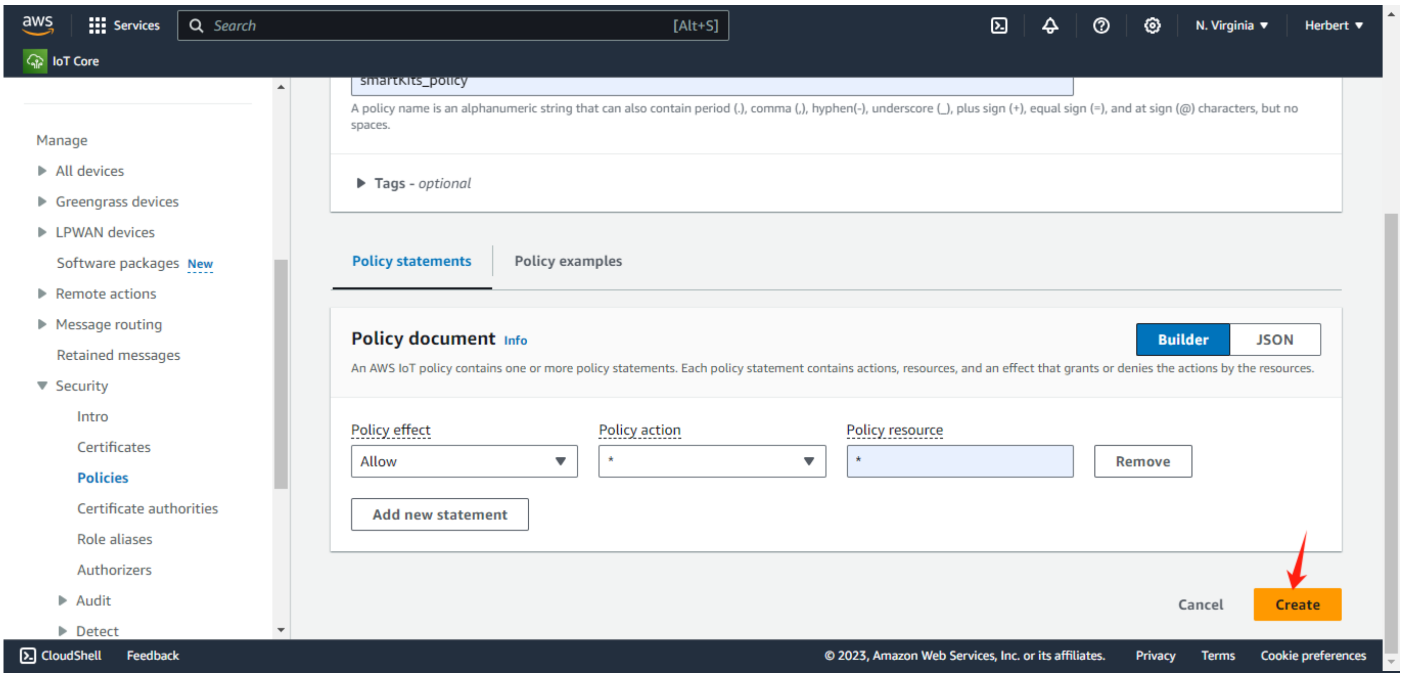
AWS IoT Core policies allow you to manage access to the AWS IoT Core data plane operations.

Policy properties

AWS IoT Core supports named policies so that many identities can reference the same policy document.

Policy name: smartKits_policy

A policy name is an alphanumeric string that can also contain period (.), comma (,), hyphen(-), underscore (_), plus sign (+), equal sign (=), and at sign (@) characters, but no spaces.


smartKits_policy

A policy name is an alphanumeric string that can also contain period (.), comma (,), hyphen(-), underscore (_), plus sign (+), equal sign (=), and at sign (@) characters, but no spaces.

▶ Tags - *optional*

Policy statements | Policy examples

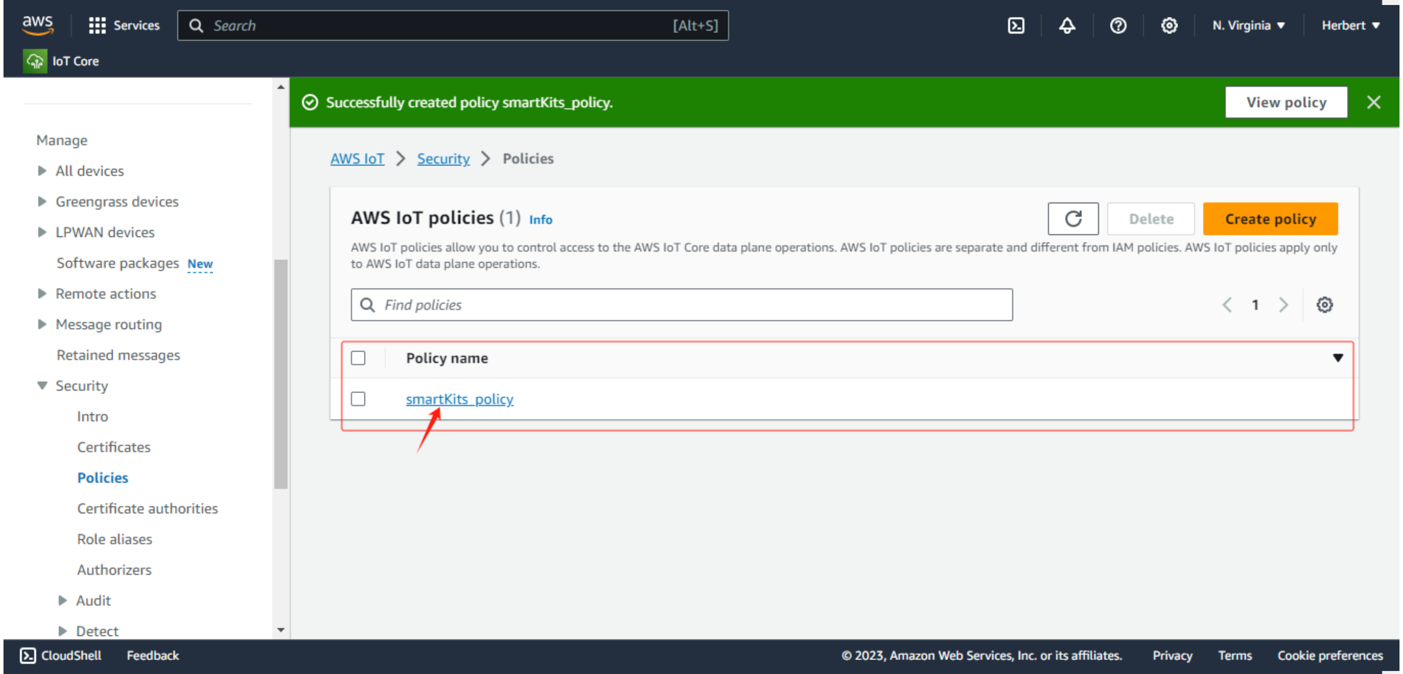
Policy document Info — Builder | JSON

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

Policy effect	Policy action	Policy resource	
Allow	*	*	Remove

Add new statement

Cancel | Create


Successfully created policy smartKits_policy. — View policy

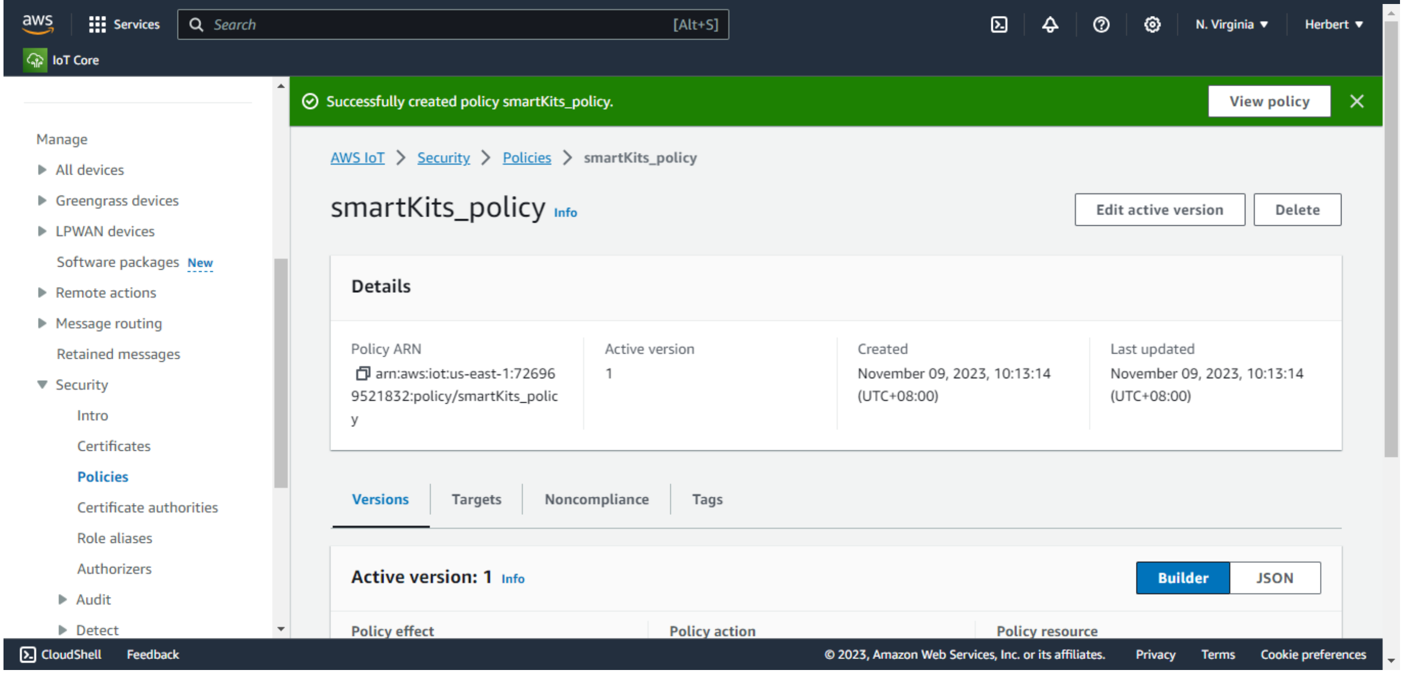
AWS IoT > Security > Policies

AWS IoT policies (1) Info — Delete | Create policy

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

Find policies

Policy name
smartKits_policy


Successfully created policy smartKits_policy. — View policy

AWS IoT > Security > Policies > smartKits_policy

smartKits_policy Info

Edit active version | Delete

Details

Policy ARN	Active version	Created	Last updated
arn:aws:iot:us-east-1:726969521832:policy/smartKits_policy	1	November 09, 2023, 10:13:14 (UTC+08:00)	November 09, 2023, 10:13:14 (UTC+08:00)

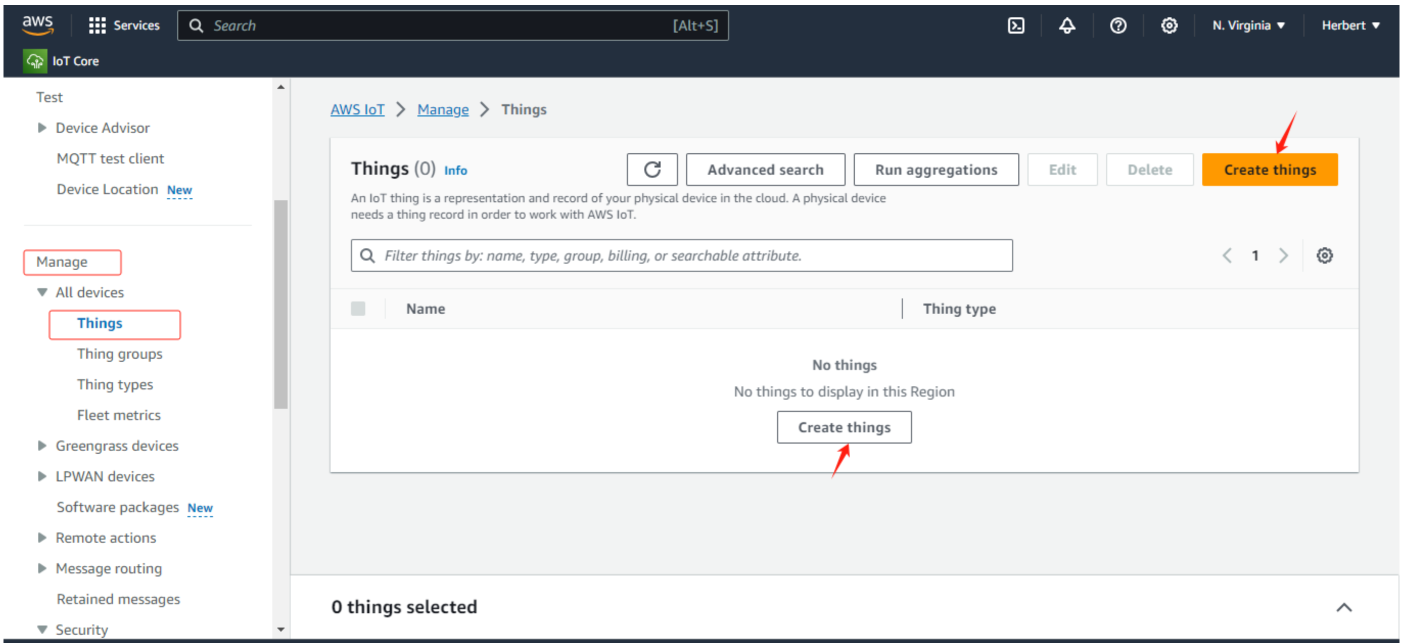
Versions | Targets | Noncompliance | Tags

Active version: 1 Info — Builder | JSON

Policy effect | Policy action | Policy resource

3.3 Add Tings

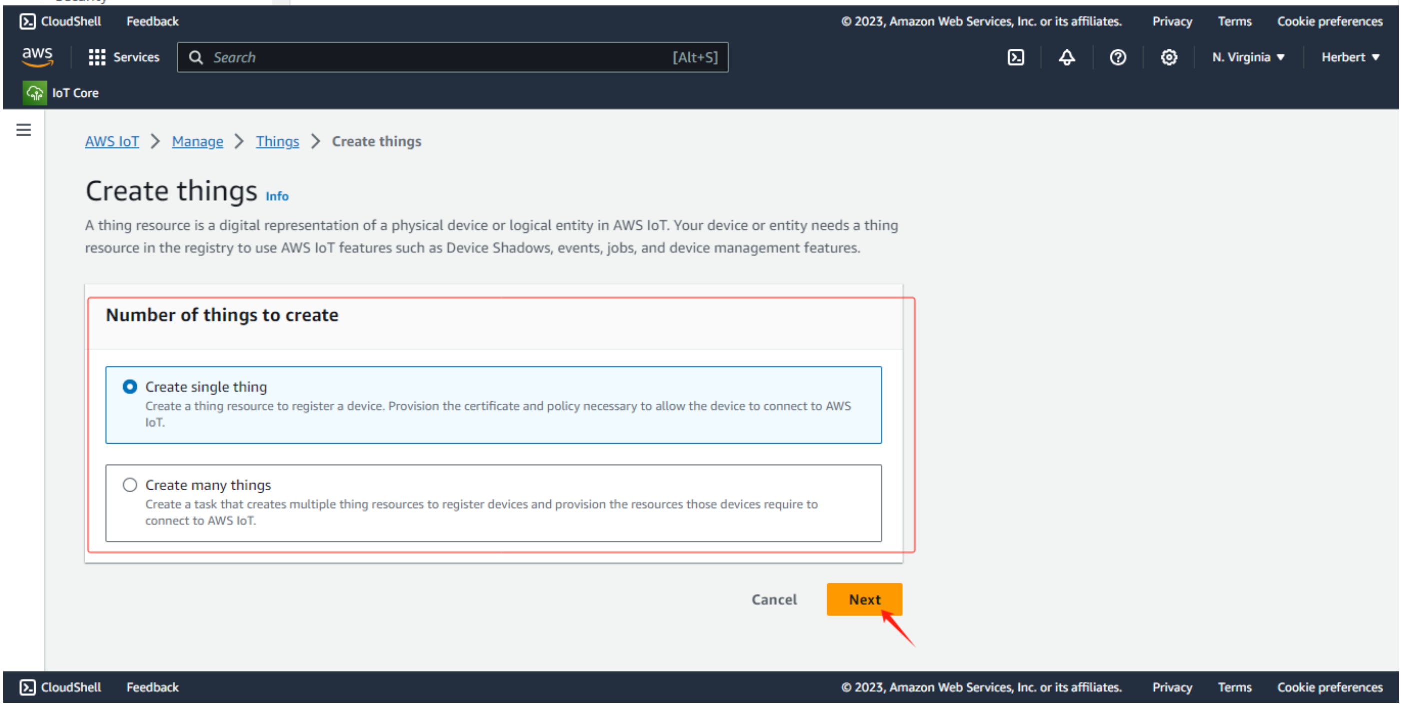
Go to the Add/create things by "Manage → All devices → Things" in the control panel on the left side of the AWS IoT page, click "Create things", and create a single thing or multiple things according to actual requirements; Take the following example of smartKit001, and then fill in and select the specified things properties (optional); to select other configurations based on actual application requirements, as shown below.



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

No things to display in this Region

© 2023, Amazon Web Services, Inc. or its affiliates.

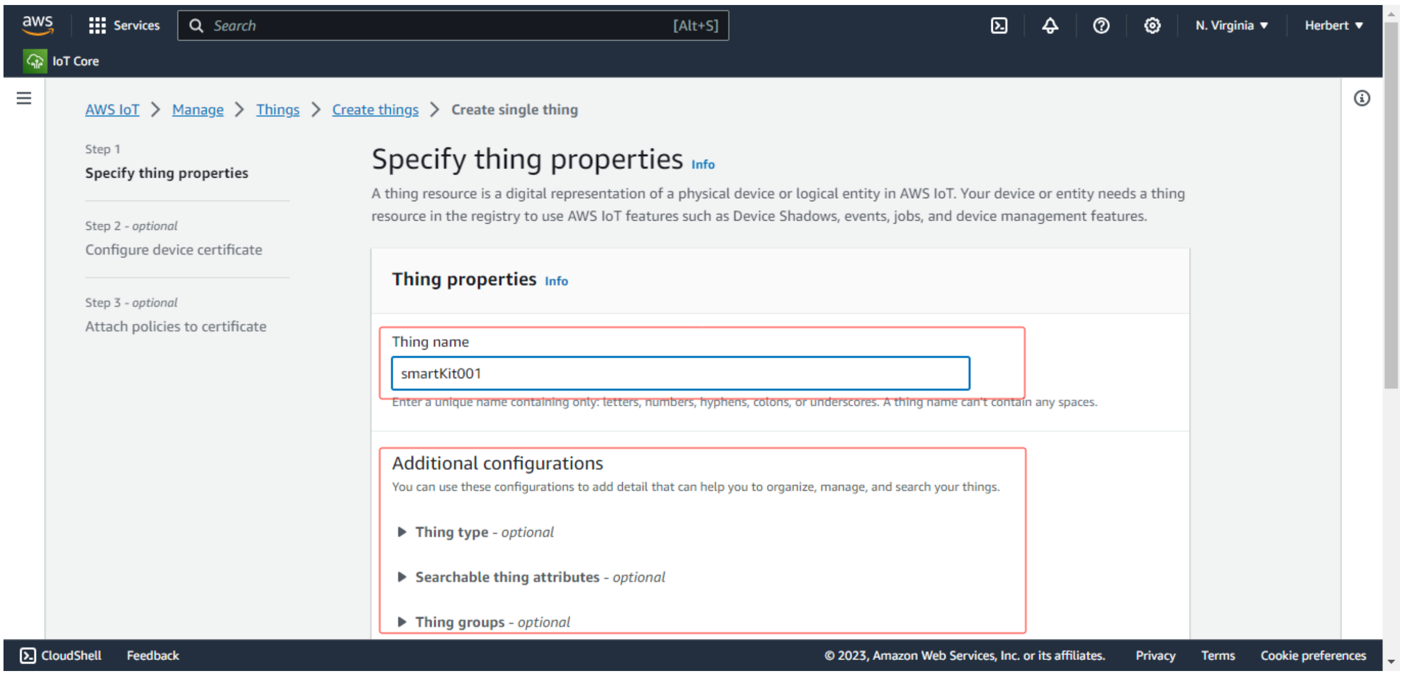


A thing resource is a digital representation of a physical device or logical entity in AWS IoT. Your device or entity needs a thing resource in the registry to use AWS IoT features such as Device Shadows, events, jobs, and device management features.

● Create single thing
Create a thing resource to register a device. Provision the certificate and policy necessary to allow the device to connect to AWS IoT.

○ Create many things
Create a task that creates multiple thing resources to register devices and provision the resources those devices require to connect to AWS IoT.

© 2023, Amazon Web Services, Inc. or its affiliates.


AWS IoT > Manage > Things > Create things > Create single thing

Step 1
Specify thing properties

Step 2 - optional
Configure device certificate

Step 3 - optional
Attach policies to certificate

Specify thing properties

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

Thing properties

Thing name: smartKit001

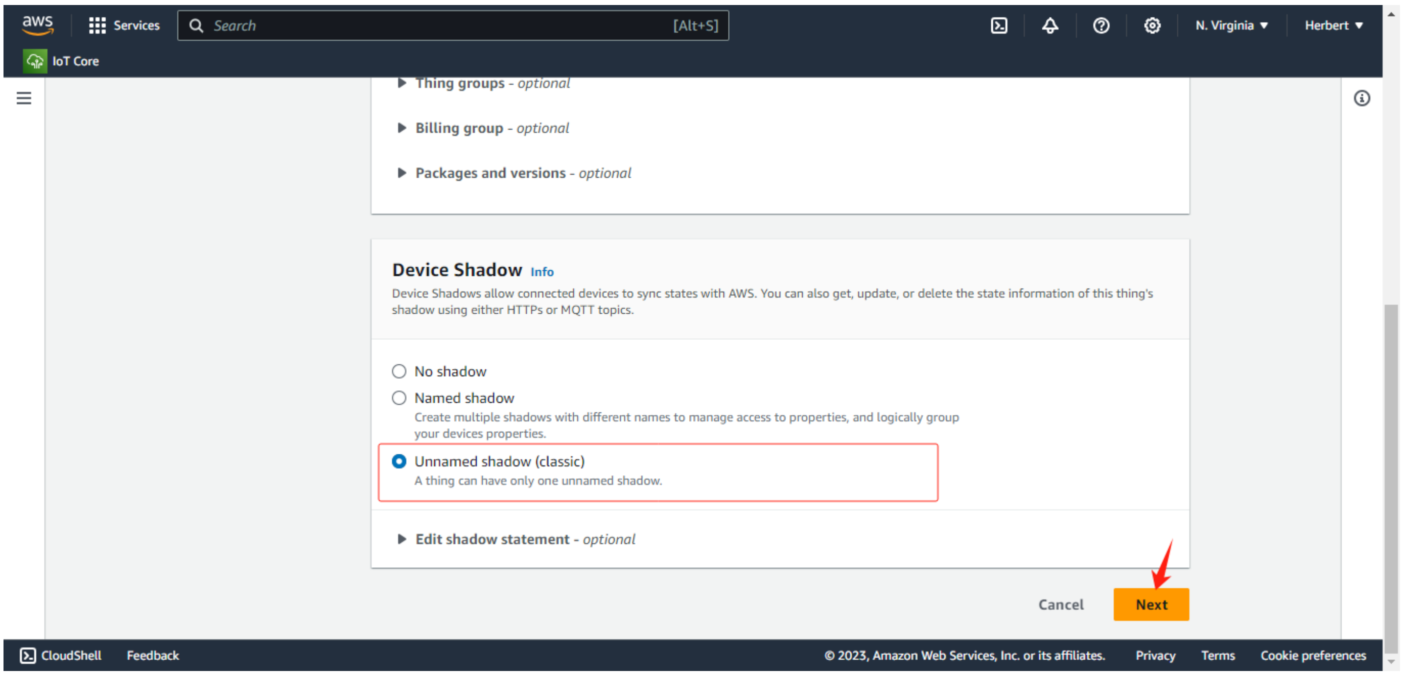
Enter a unique name containing only: letters, numbers, hyphens, colons, or underscores. A thing name can't contain any spaces.

Additional configurations

You can use these configurations to add detail that can help you to organize, manage, and search your things.

- Thing type - optional
- Searchable thing attributes - optional
- Thing groups - optional

Select whether to add "Device Shadow" according to the actual application requirements.To facilitate subsequent debugging, currently select "Unnamed Shadow (Classic)";


- Thing groups - optional
- Billing group - optional
- Packages and versions - optional

Device Shadow

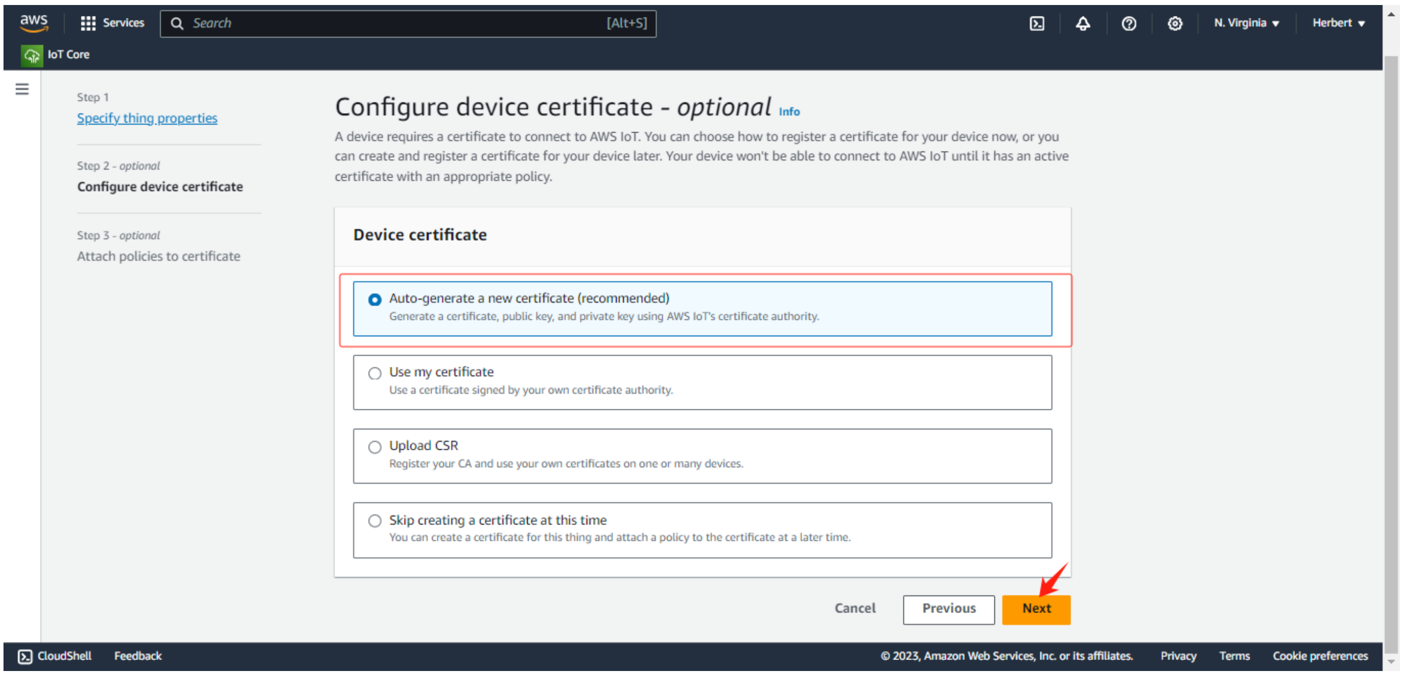
Device Shadows allow connected devices to sync states with AWS. You can also get, update, or delete the state information of this thing's shadow using either HTTPs or MQTT topics.

- ○ No shadow
- ○ Named shadow
  Create multiple shadows with different names to manage access to properties, and logically group your devices properties.
- [✓ Unnamed shadow (classic)]
  A thing can have only one unnamed shadow.

Edit shadow statement - optional

Cancel | Next

You need to configure the device certificate based on the different mode of certificate. Let's take “Auto-Generate a New certificate(recommended)” as an example.



Step 1
Specify thing properties

Step 2 - *optional*
Configure device certificate

Step 3 - *optional*
Attach policies to certificate

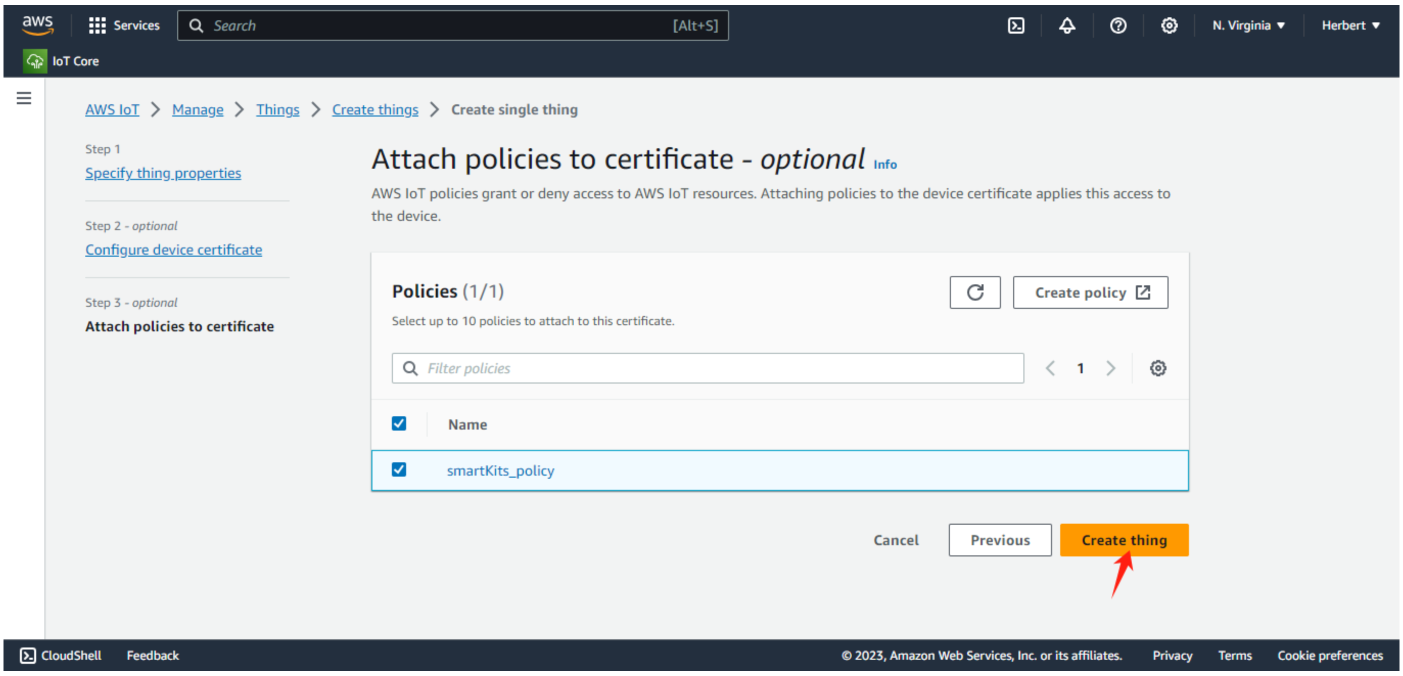
Configure device certificate - *optional* Info

A device requires a certificate to connect to AWS IoT. You can choose how to register a certificate for your device now, or you can create and register a certificate for your device later. Your device won't be able to connect to AWS IoT until it has an active certificate with an appropriate policy.

Device certificate

- [✓ Auto-generate a new certificate (recommended)] Generate a certificate, public key, and private key using AWS IoT's certificate authority.
- Use my certificate — Use a certificate signed by your own certificate authority.
- Upload CSR — Register your CA and use your own certificates on one or many devices.
- Skip creating a certificate at this time — You can create a certificate for this thing and attach a policy to the certificate at a later time.

Attach a policy to the current thing, select the policy created above, and obtain the relevant certificate; As shown below;



AWS IoT > Manage > Things > Create things > Create single thing

Step 1
Specify thing properties

Step 2 - *optional*
Configure device certificate

Step 3 - *optional*
Attach policies to certificate

Attach policies to certificate - *optional* Info

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

Policies (1/1)

Select up to 10 policies to attach to this certificate.

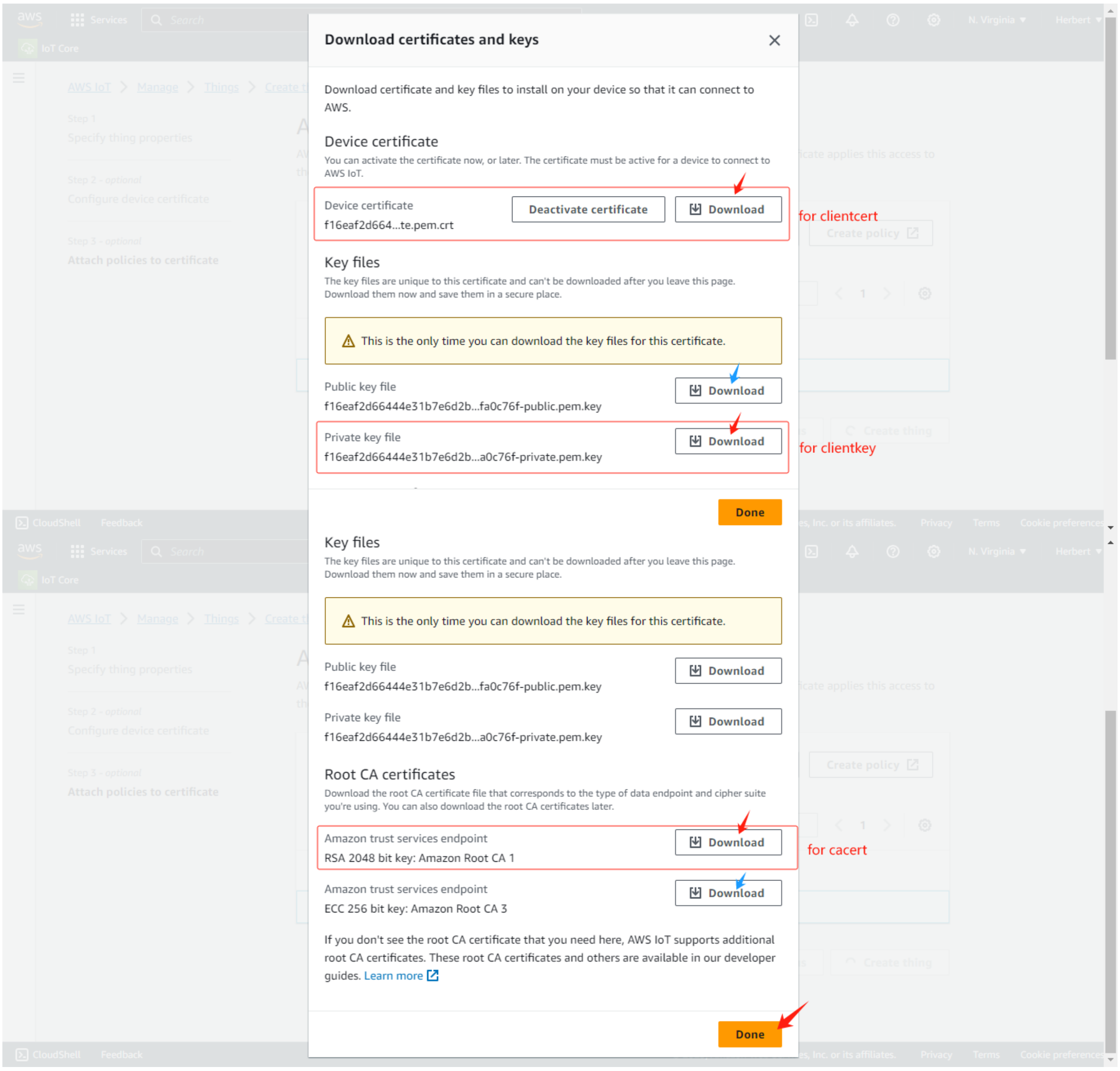
✓	Name
✓	smartKits_policy

On the "Download Certificates and Keys" page, download the CA certificate and key files automatically generated by the platform and save it.

xxxxxxxxxxx-certificate.pem for clientcert （Client Certificate File）；

xxxxxxxxxxx-private.pem for clientkey （Client Key File）；

AmazonRootCA1.pem for cacert （CA File）



Download certificates and keys

Download certificate and key files to install on your device so that it can connect to AWS.

Device certificate

You can activate the certificate now, or later. The certificate must be active for a device to connect to AWS IoT.

Device certificate
f16eaf2d664...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
f16eaf2d66444e31b7e6d2b...fa0c76f-public.pem.key

Private key file
f16eaf2d66444e31b7e6d2b...a0c76f-private.pem.key

Root CA certificates

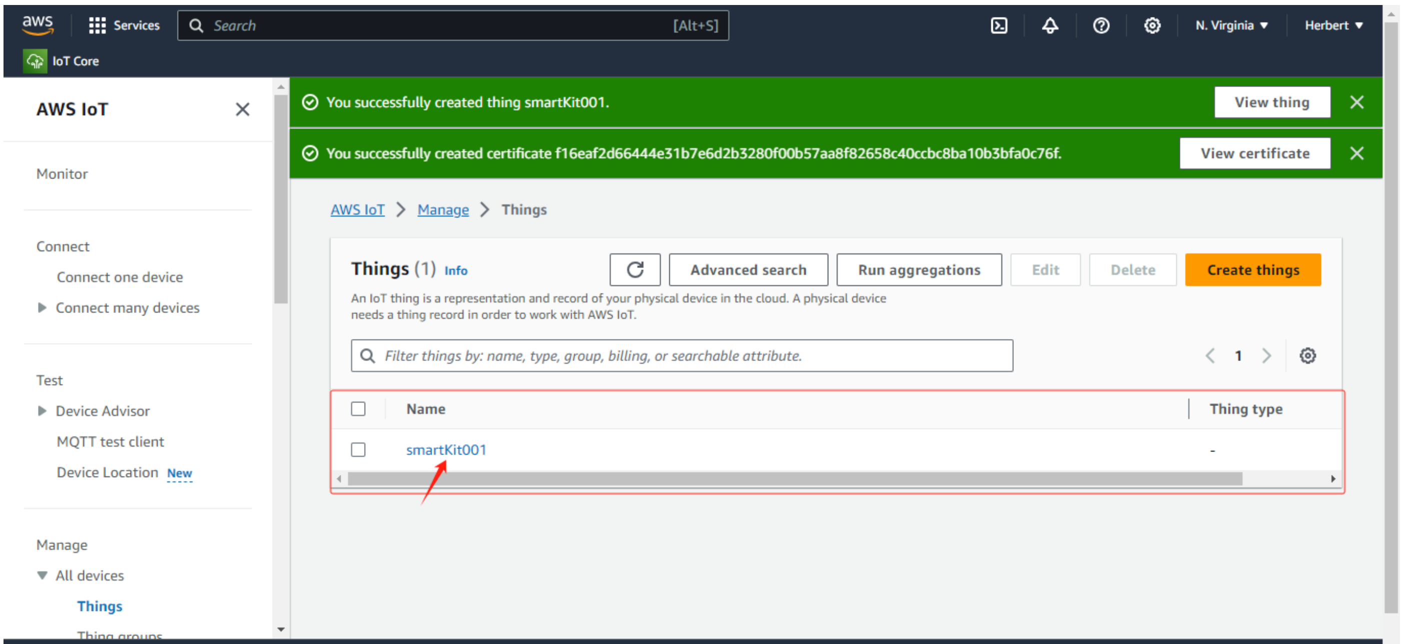
Download the root CA certificate file that corresponds to the type of data endpoint and cipher suite you're using. You can also download the root CA certificates later.

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

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

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

You can click "smartKit001" to view the thing details;


You successfully created thing smartKit001.

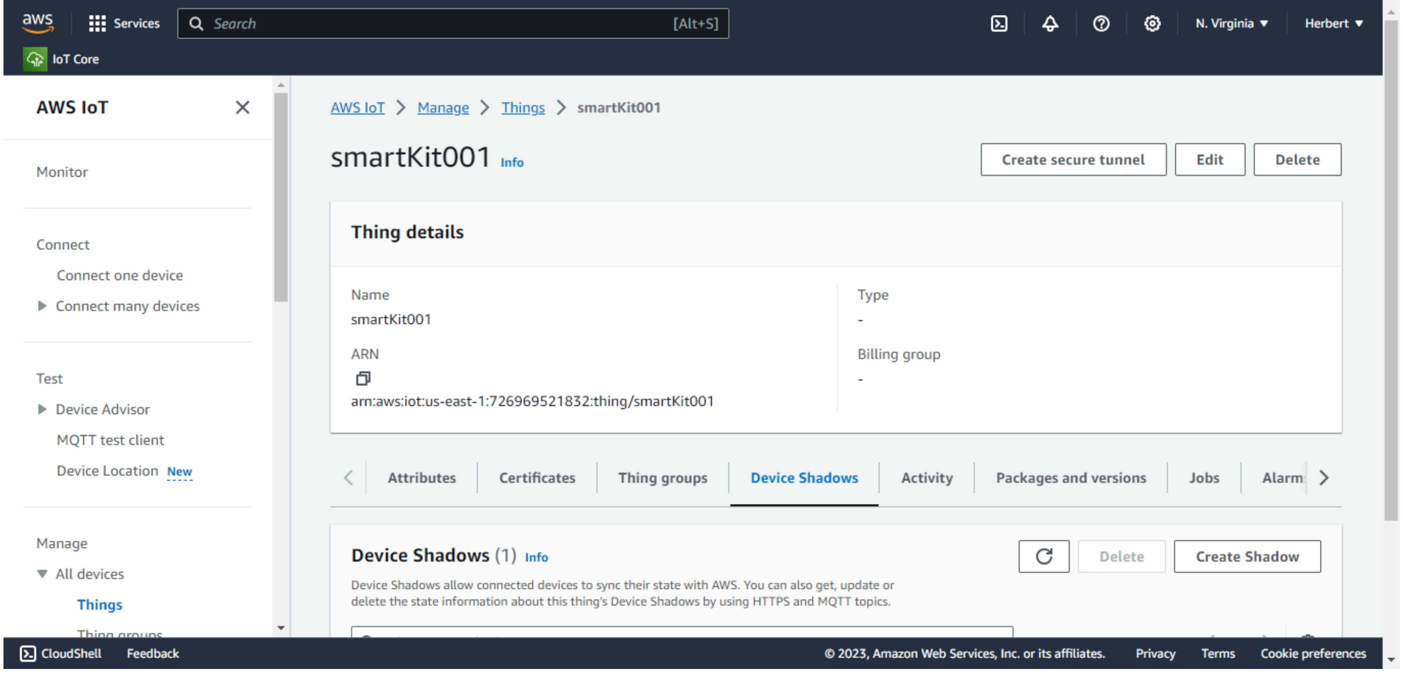
You successfully created certificate f16eaf2d66444e31b7e6d2b3280f00b57aa8f82658c40ccbc8ba10b3bfa0c76f.

AWS IoT > Manage > Things

Things (1) Info

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

Name	Thing type
smartKit001	-


AWS IoT > Manage > Things > smartKit001

smartKit001 Info

Thing details

Name: smartKit001

Type: -

ARN: arn:aws:iot:us-east-1:726969521832:thing/smartKit001

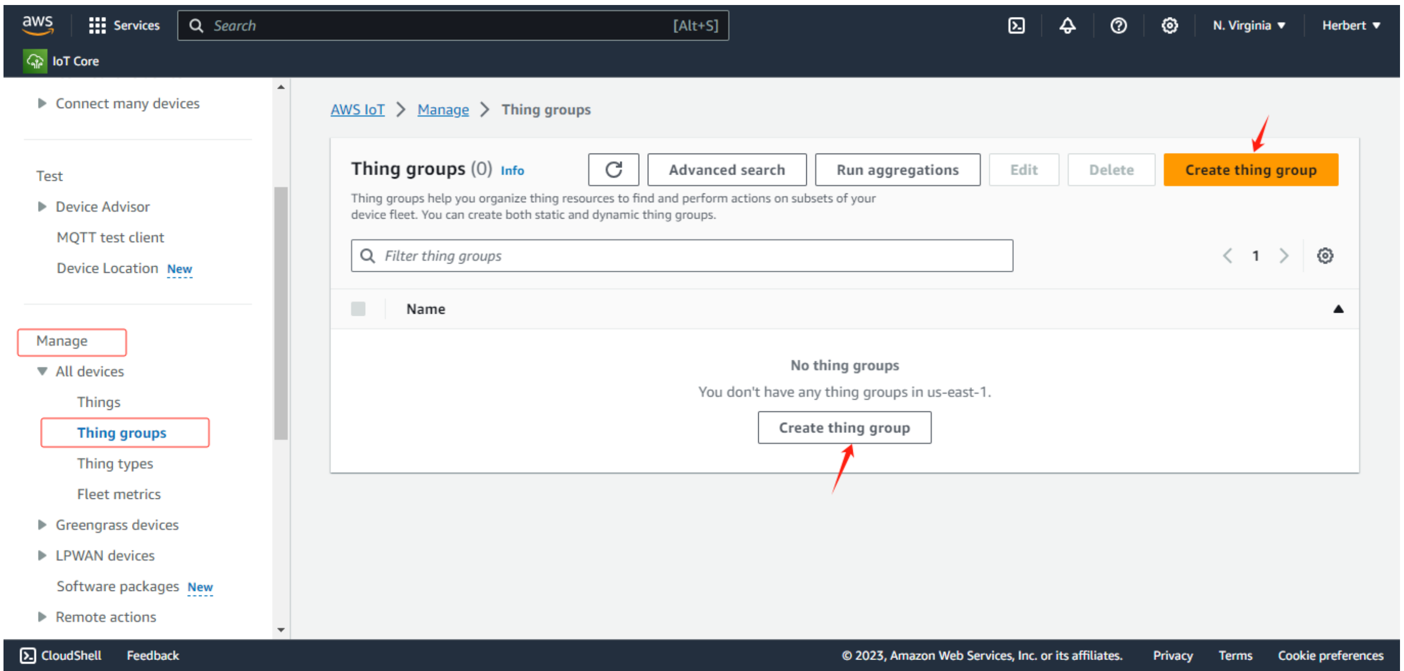
Billing group: -

Device Shadows (1) Info

Device Shadows allow connected devices to sync their state with AWS. You can also get, update or delete the state information about this thing's Device Shadows by using HTTPS and MQTT topics.

3.4 Add Ting Groups(Options)

As shown in the figure below, you can enter the thing groups configuration page through "Manage → All Devices → thing Groups" in the control panel on the left side of the AWS IoT, click "Create thing Group",then to select the thing group type, fill in the corresponding the name of thing group, and configure other options according to requirements, and then to click "Create thing Group".

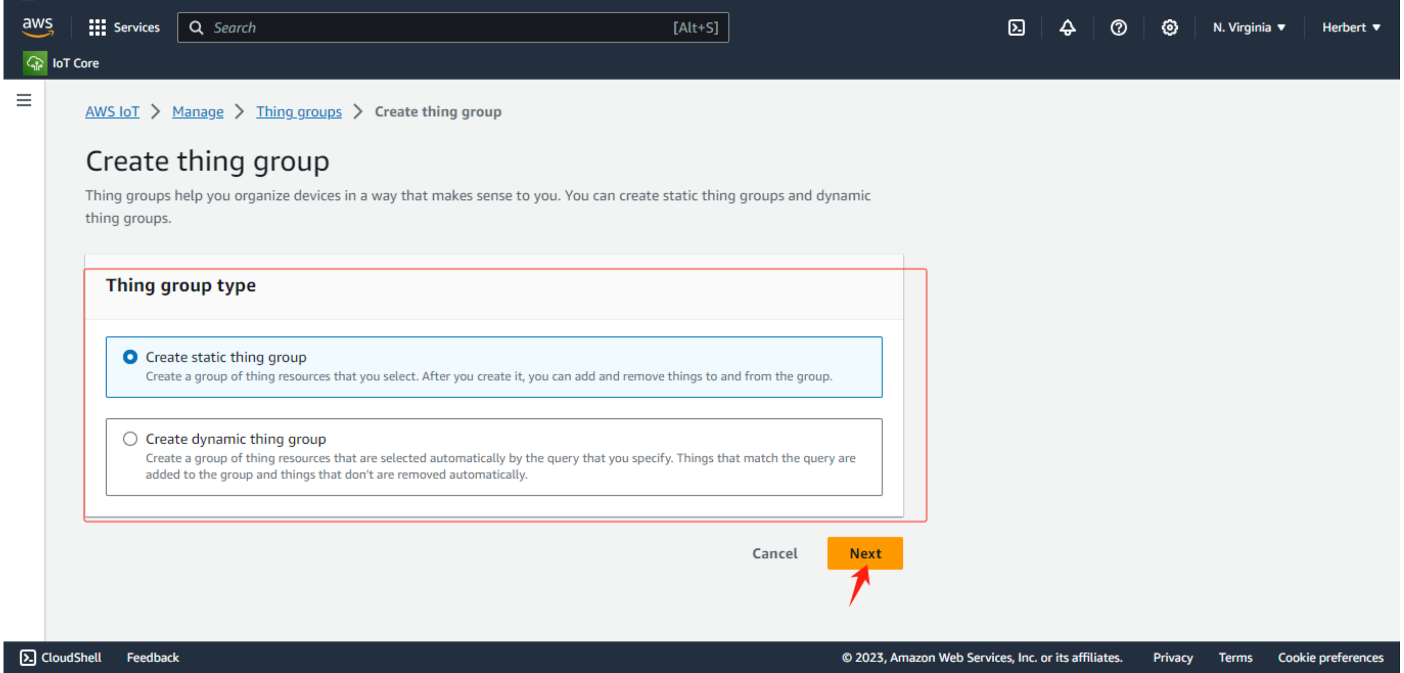


Thing groups (0) Info

Thing groups help you organize thing resources to find and perform actions on subsets of your device fleet. You can create both static and dynamic thing groups.

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

© 2023, Amazon Web Services, Inc. or its affiliates.



Create thing group

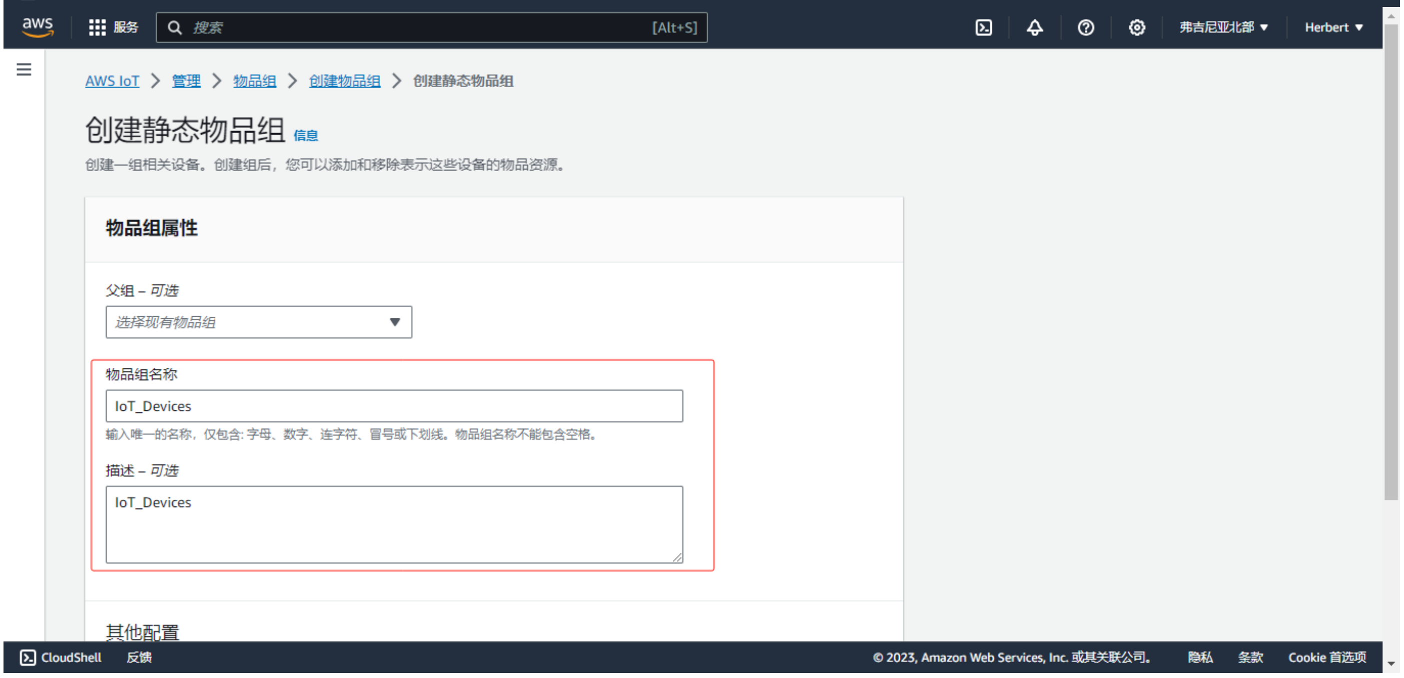
Thing groups help you organize devices in a way that makes sense to you. You can create static thing groups and dynamic thing groups.

Thing group type

[✓ Create static thing group]
Create a group of thing resources that you select. After you create it, you can add and remove things to and from the group.

Create dynamic thing group
Create a group of thing resources that are selected automatically by the query that you specify. Things that match the query are added to the group and things that don't are removed automatically.

© 2023, Amazon Web Services, Inc. or its affiliates.



创建静态物品组 信息

创建一组相关设备。创建组后，您可以添加和移除表示这些设备的物品资源。

物品组属性

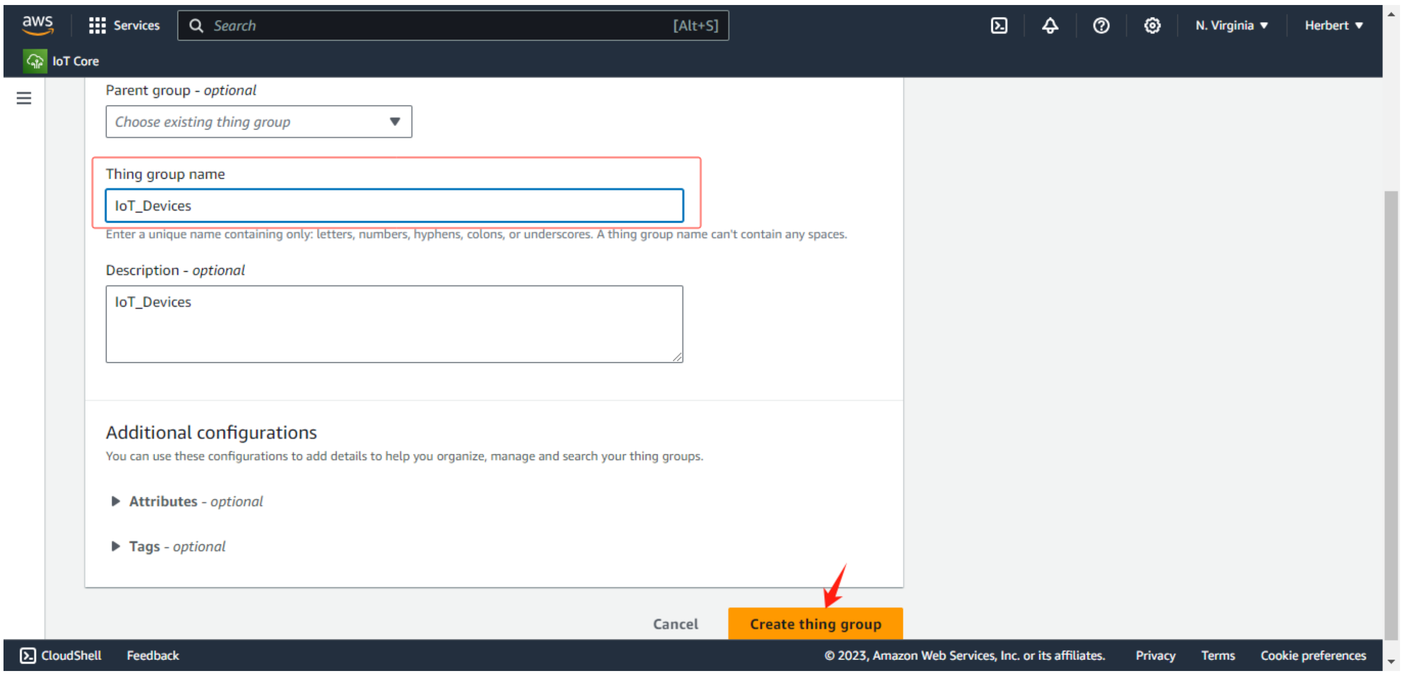
父组 – 可选
选择现有物品组

物品组名称
IoT_Devices
输入唯一的名称，仅包含: 字母、数字、连字符、冒号或下划线。物品组名称不能包含空格。

描述 – 可选
IoT_Devices

其他配置

© 2023, Amazon Web Services, Inc. 或其关联公司。


Parent group - *optional*

Choose existing thing group

Thing group name

IoT_Devices

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

Description - *optional*

IoT_Devices

Additional configurations

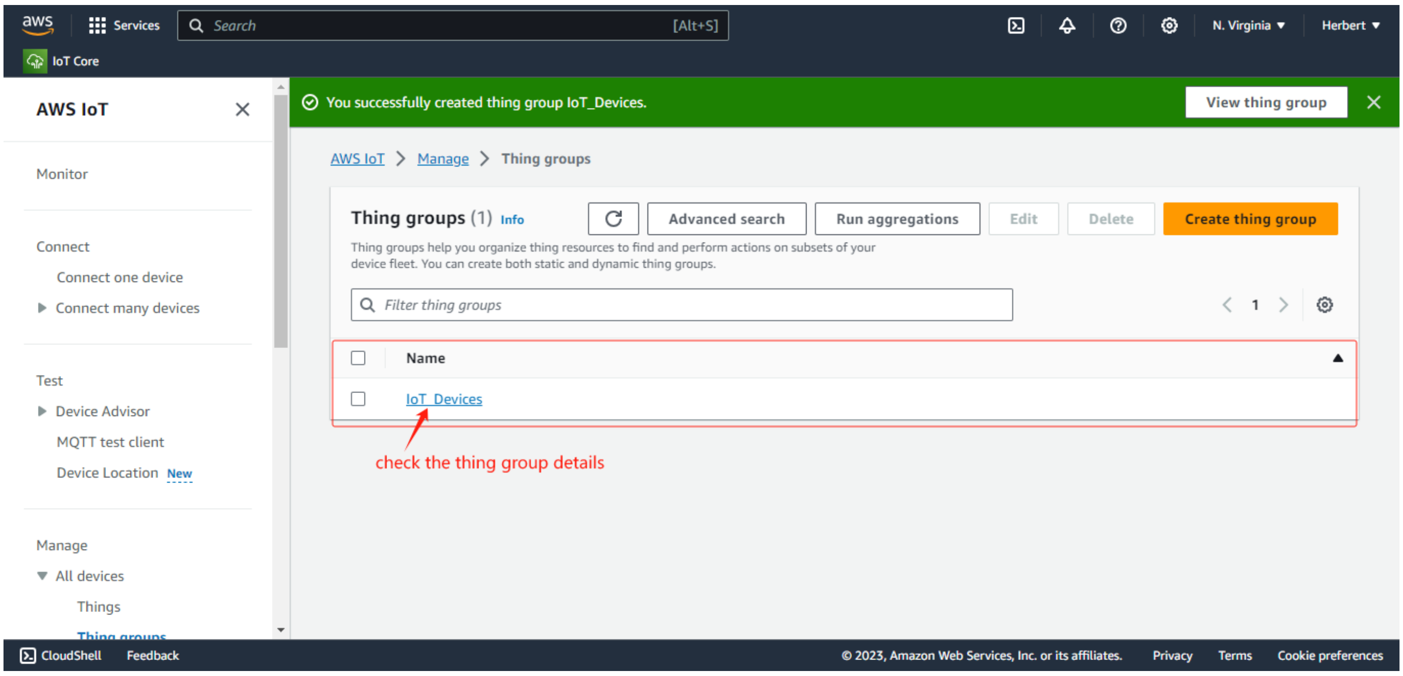
You can use these configurations to add details to help you organize, manage and search your thing groups.

▶ **Attributes** - *optional*

▶ **Tags** - *optional*

Cancel | Create thing group

Click on the created thing group to view the details, as shown below;


You successfully created thing group IoT_Devices.

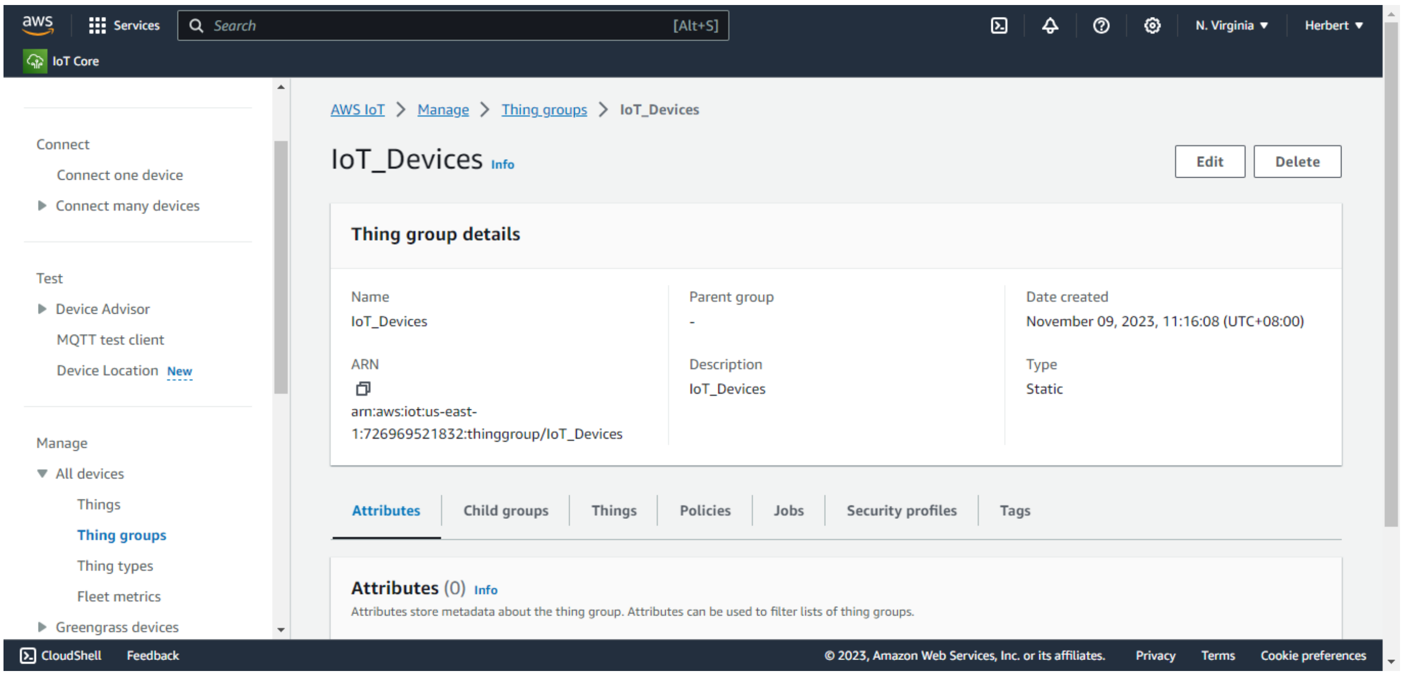
AWS IoT > Manage > Thing groups

Thing groups (1) Info

Thing groups help you organize thing resources to find and perform actions on subsets of your device fleet. You can create both static and dynamic thing groups.

Name
IoT_Devices

check the thing group details



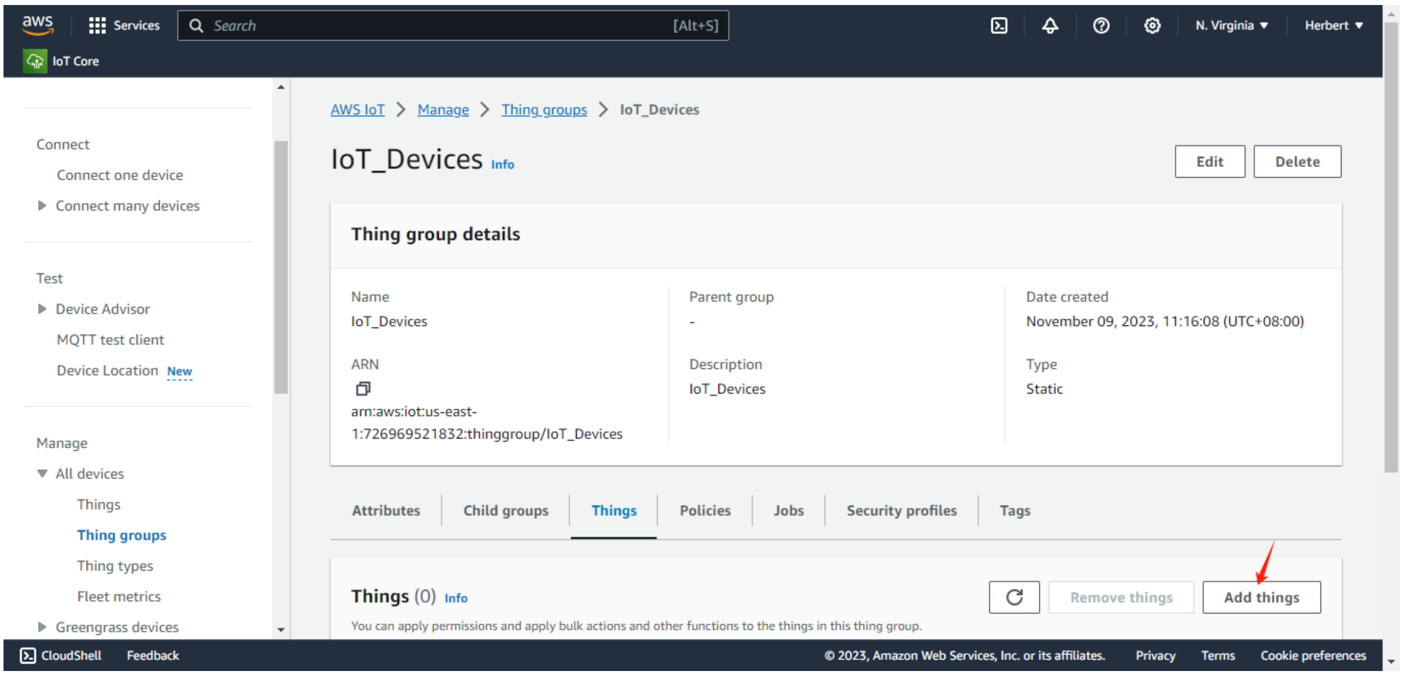
Thing group details

Name	Parent group	Date created
IoT_Devices	-	November 09, 2023, 11:16:08 (UTC+08:00)
ARN	Description	Type
arn:aws:iot:us-east-1:726969521832:thinggroup/IoT_Devices	IoT_Devices	Static

Attributes (0) Info

Attributes store metadata about the thing group. Attributes can be used to filter lists of thing groups.

By the configuration item in the thing group details, you can associate the above created "Policies" and "Things", as shown in the following figure;

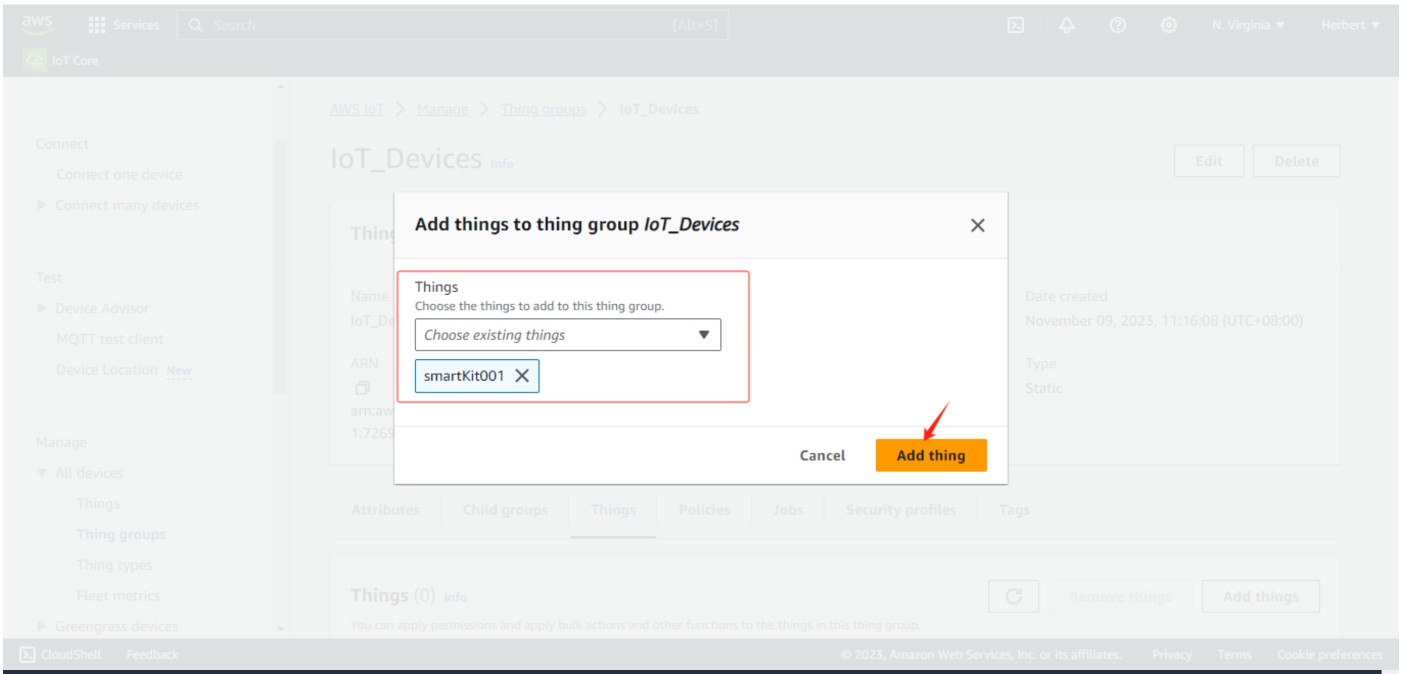


Thing group details

Name	Parent group	Date created
IoT_Devices	-	November 09, 2023, 11:16:08 (UTC+08:00)
ARN	Description	Type
arn:aws:iot:us-east-1:726969521832:thinggroup/IoT_Devices	IoT_Devices	Static

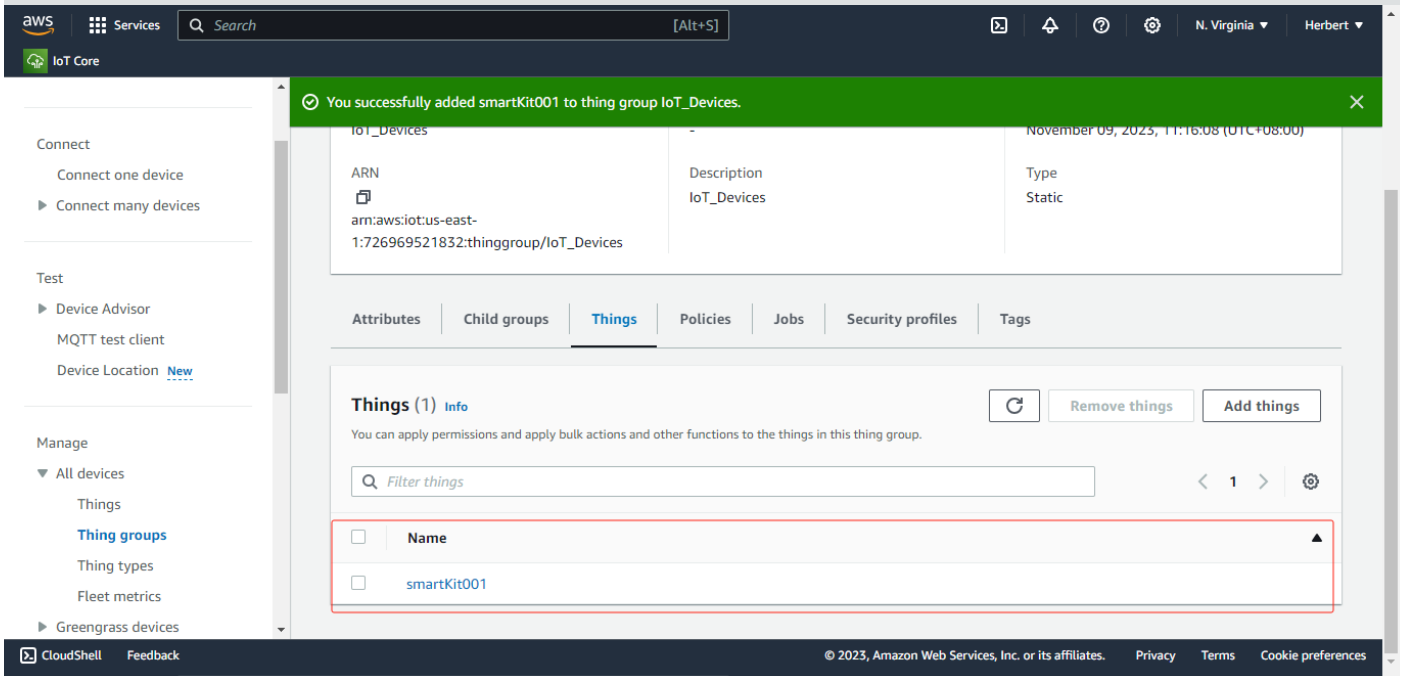
Things (0) Info

You can apply permissions and apply bulk actions and other functions to the things in this thing group.


Add things to thing group *IoT_Devices*

Things
Choose the things to add to this thing group.
Choose existing things
smartKit001

Cancel | Add thing


You successfully added smartKit001 to thing group IoT_Devices.

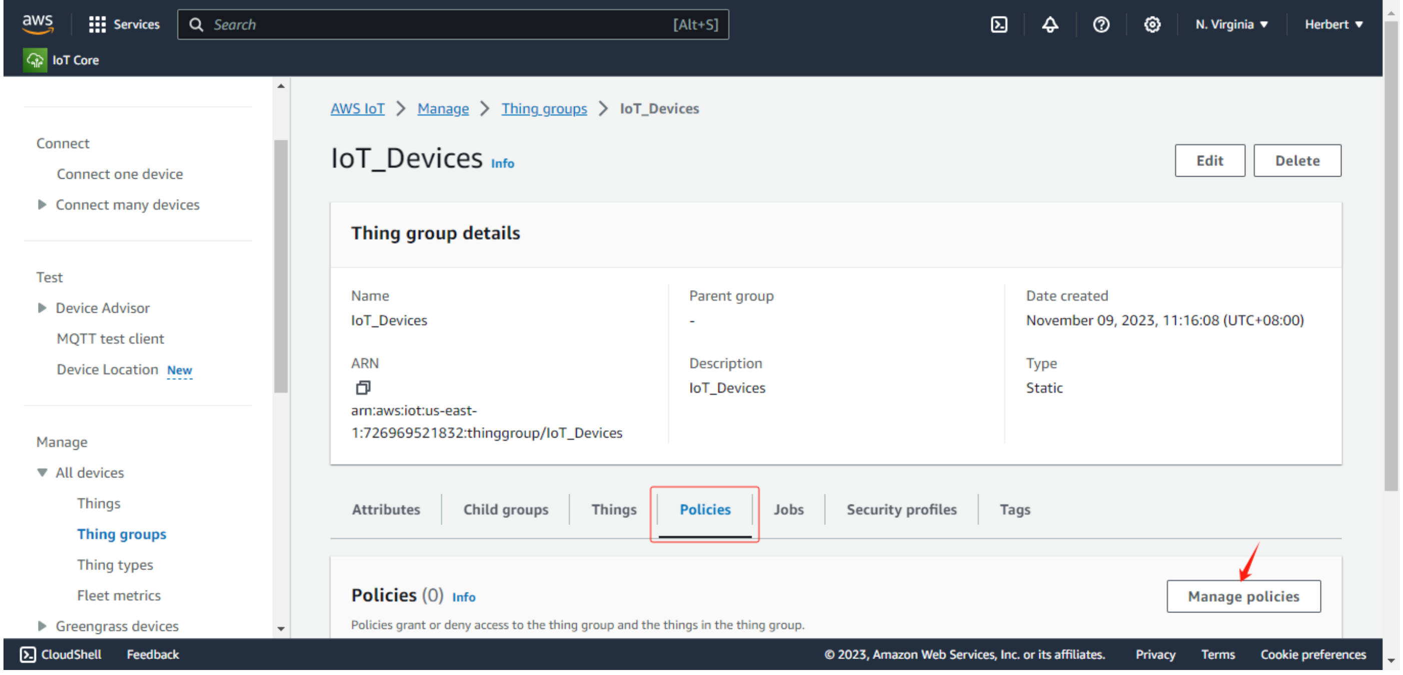
ARN
arn:aws:iot:us-east-1:726969521832:thinggroup/IoT_Devices

Description
IoT_Devices

Type
Static

Things (1)
You can apply permissions and apply bulk actions and other functions to the things in this thing group.

Name
smartKit001


AWS IoT > Manage > Thing groups > IoT_Devices

IoT_Devices

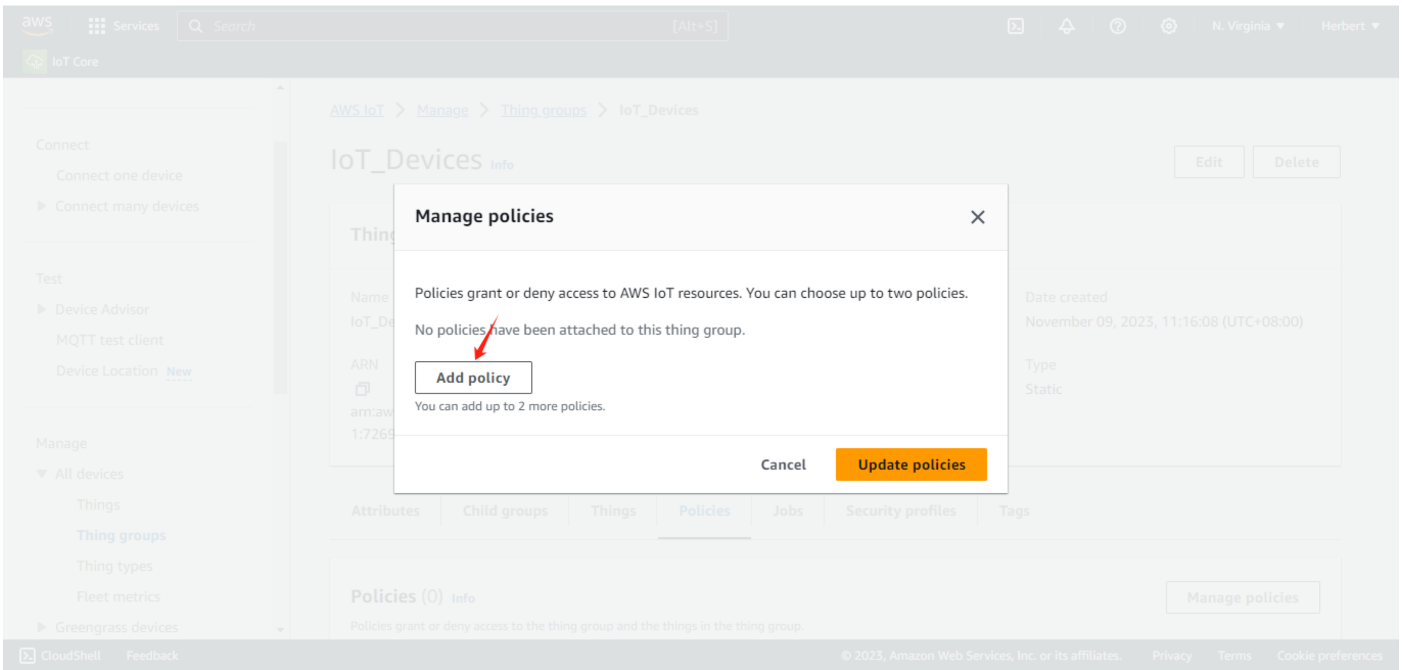
Thing group details

Name IoT_Devices	Parent group -	Date created November 09, 2023, 11:16:08 (UTC+08:00)
ARN arn:aws:iot:us-east-1:726969521832:thinggroup/IoT_Devices	Description IoT_Devices	Type Static

Attributes | Child groups | Things | Policies | Jobs | Security profiles | Tags

Policies (0)
Policies grant or deny access to the thing group and the things in the thing group.

Manage policies


AWS IoT > Manage > Thing groups > IoT_Devices

IoT_Devices

Manage policies

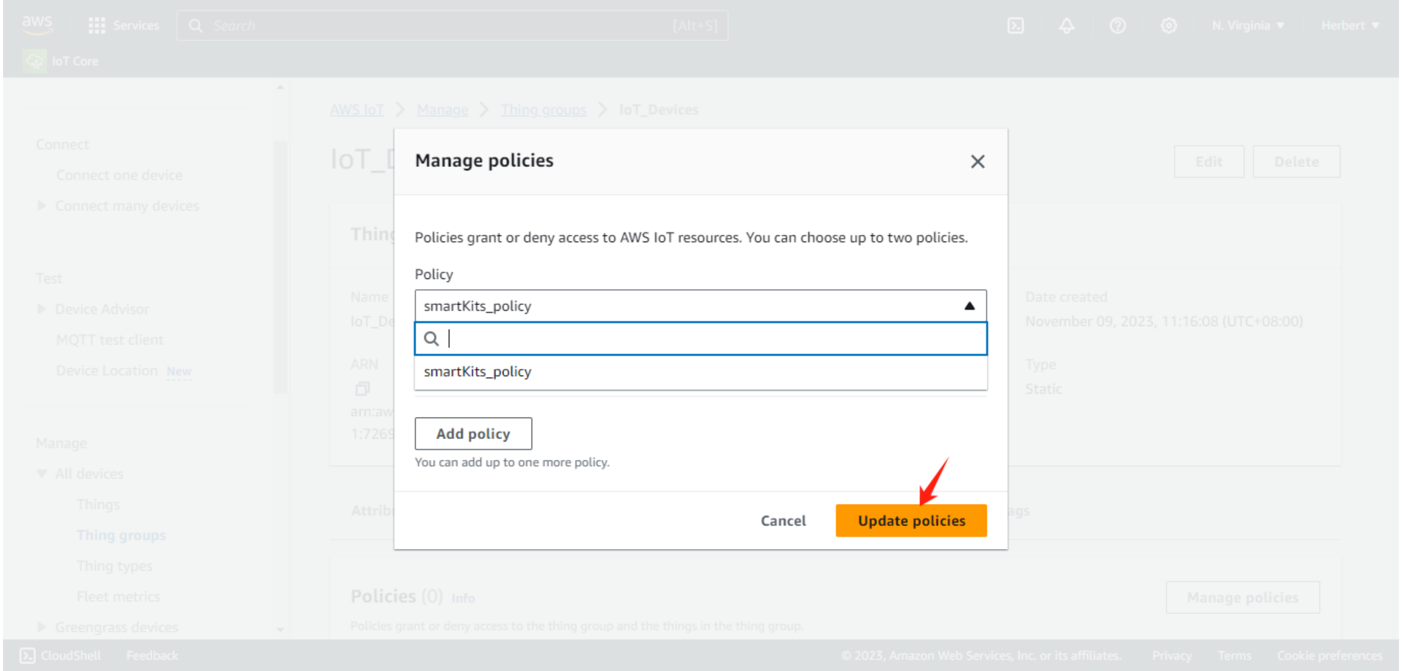
Policies grant or deny access to AWS IoT resources. You can choose up to two policies.

No policies have been attached to this thing group.

Add policy

You can add up to 2 more policies.

Cancel | Update policies


Manage policies

Policies grant or deny access to AWS IoT resources. You can choose up to two policies.

Policy

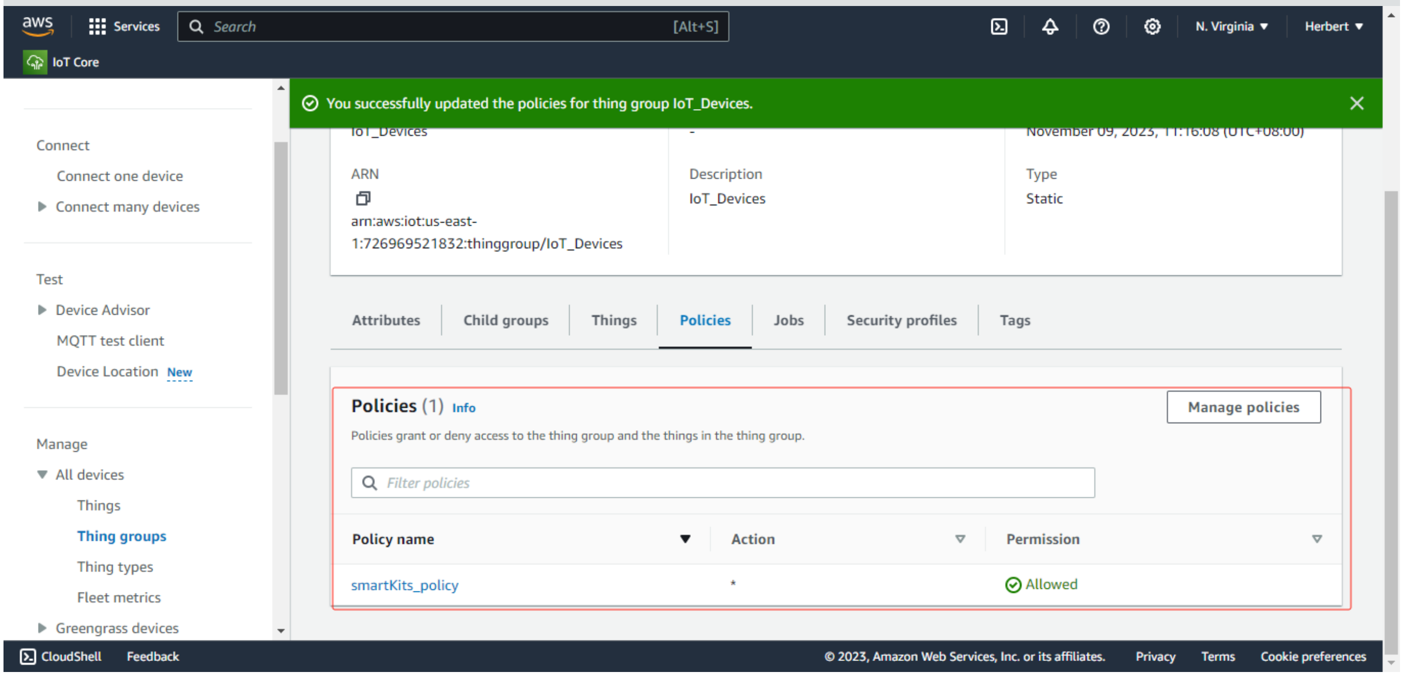
smartKits_policy

smartKits_policy

Add policy

You can add up to one more policy.

Cancel | Update policies


You successfully updated the policies for thing group IoT_Devices.

ARN
arn:aws:iot:us-east-1:726969521832:thinggroup/IoT_Devices

Description
IoT_Devices

Type
Static

Attributes | Child groups | Things | Policies | Jobs | Security profiles | Tags

Policies (1) Info

Policies grant or deny access to the thing group and the things in the thing group.

Manage policies

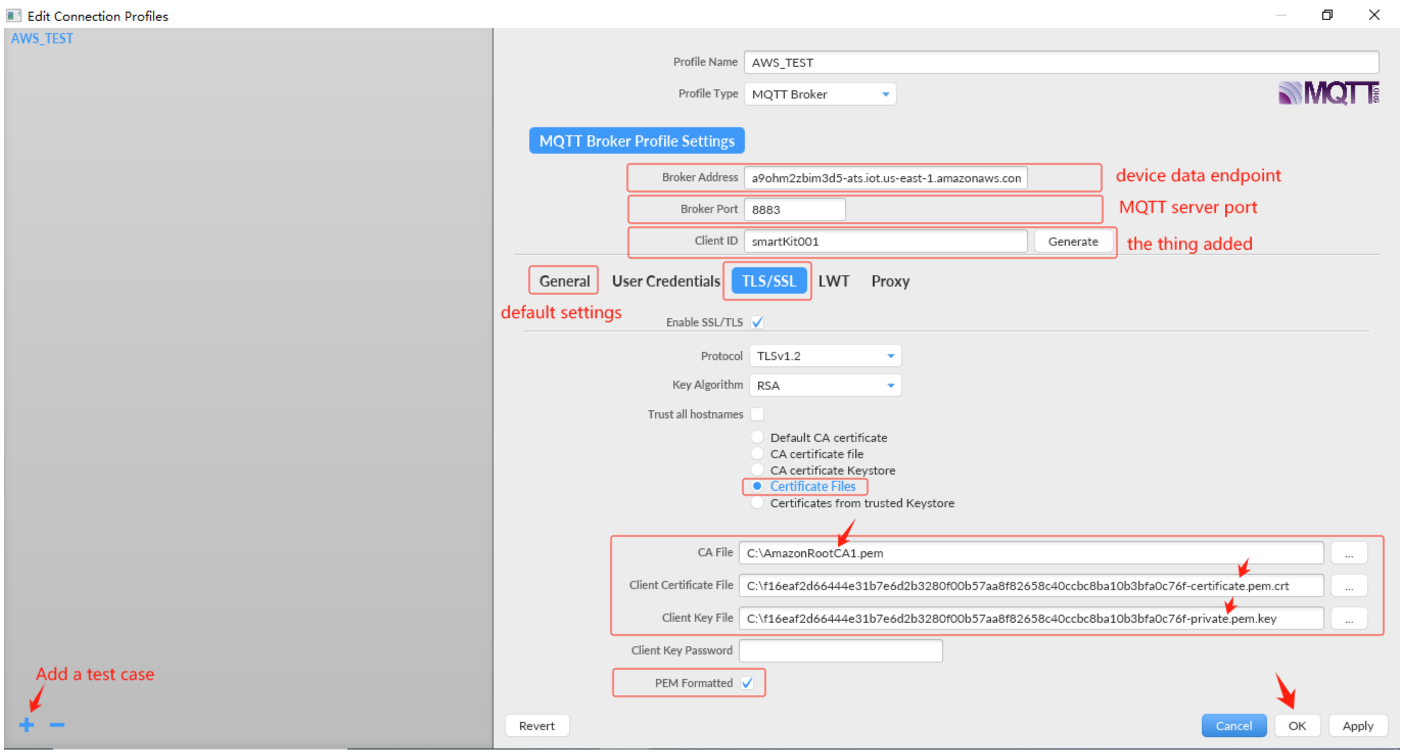
Policy name	Action	Permission
smartKits_policy	*	Allowed

四、MQTT Test

4.1 MQTT.fx Configuration and Access to AWS IoT

The following tests are based on the MQTT.fx tool：https://softblade.de/en/download-2/;

the detailed configuration as shown below;



Edit Connection Profiles

AWS_TEST

Profile Name: AWS_TEST
Profile Type: MQTT Broker

MQTT Broker Profile Settings

Broker Address: a9ohm2zbim3d5-ats.iot.us-east-1.amazonaws.con — device data endpoint
Broker Port: 8883 — MQTT server port
Client ID: smartKit001 — the thing added

General | User Credentials | TLS/SSL | LWT | Proxy

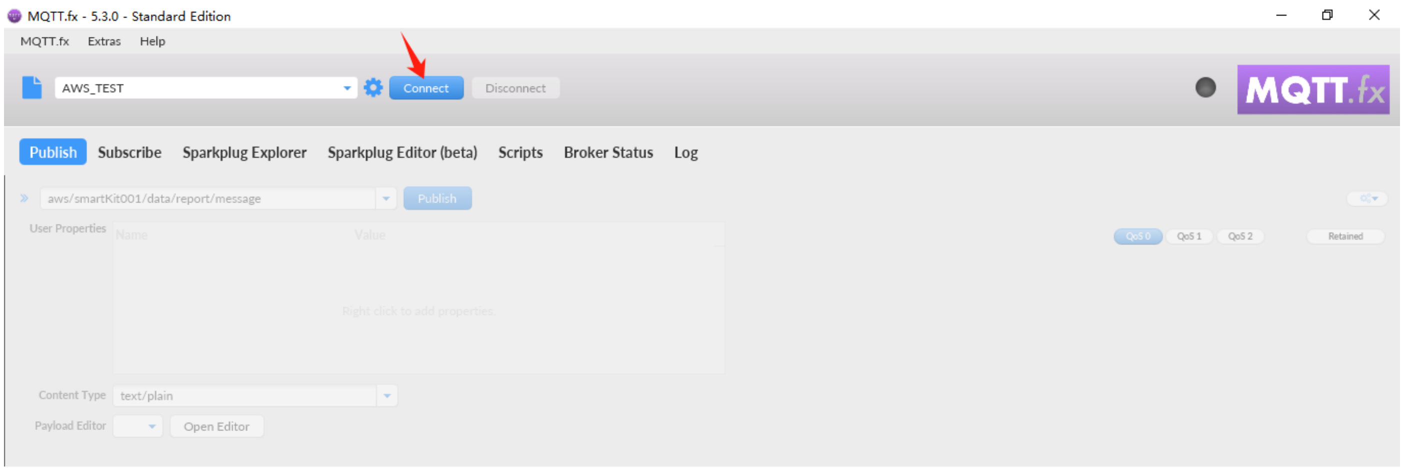
default settings

Enable SSL/TLS ✓
Protocol: TLSv1.2
Key Algorithm: RSA
Trust all hostnames ☐

- Default CA certificate
- CA certificate file
- CA certificate Keystore
- ● Certificate Files
- Certificates from trusted Keystore

CA File: C:\AmazonRootCA1.pem
Client Certificate File: C:\f16eaf2d66444e31b7e6d2b3280f00b57aa8f82658c40ccbc8ba10b3bfa0c76f-certificate.pem.crt
Client Key File: C:\f16eaf2d66444e31b7e6d2b3280f00b57aa8f82658c40ccbc8ba10b3bfa0c76f-private.pem.key
Client Key Password:
PEM Formatted ✓

Add a test case

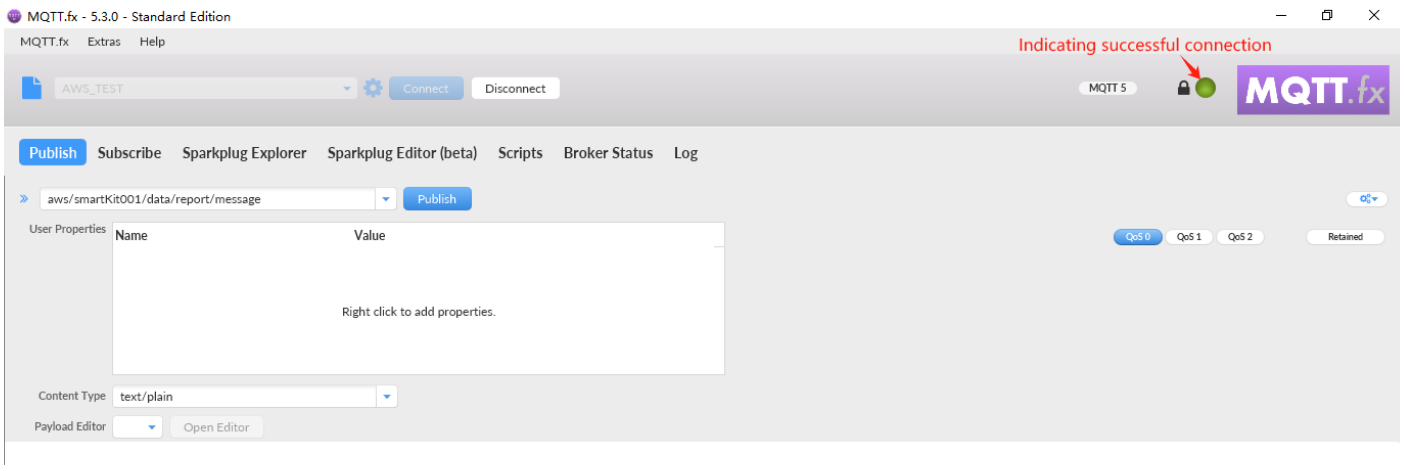


MQTT.fx - 5.3.0 - Standard Edition

AWS_TEST

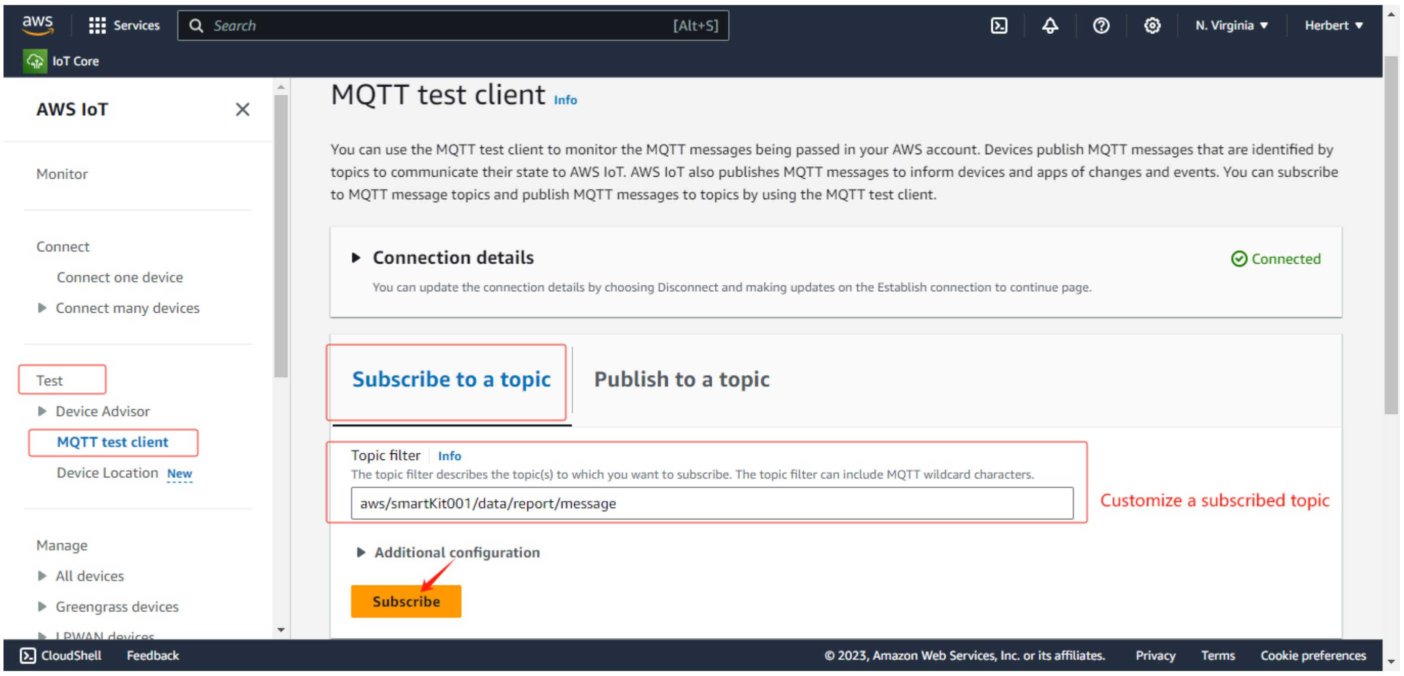
aws/smartKit001/data/report/message

Content Type: text/plain


MQTT.fx - 5.3.0 - Standard Edition
MQTT.fx Extras Help
Indicating successful connection
AWS_TEST
Connect
Disconnect
MQTT 5
MQTT.fx
Publish Subscribe Sparkplug Explorer Sparkplug Editor (beta) Scripts Broker Status Log
aws/smartKit001/data/report/message
Publish
QoS 0 QoS 1 QoS 2 Retained
User Properties
Name
Value
Right click to add properties.
Content Type
text/plain
Payload Editor
Open Editor

4.2 Client Publish

As shown below, "MQTT test client"based on AWS IoT platform,you can verifie and debug the connected terminal; "MQTT test Client" displays "Connected", customizing the relevant subscribed topic, MQtt.fx virtual terminal can publish messages to the topic, and the platform can view the messages published by the terminal;



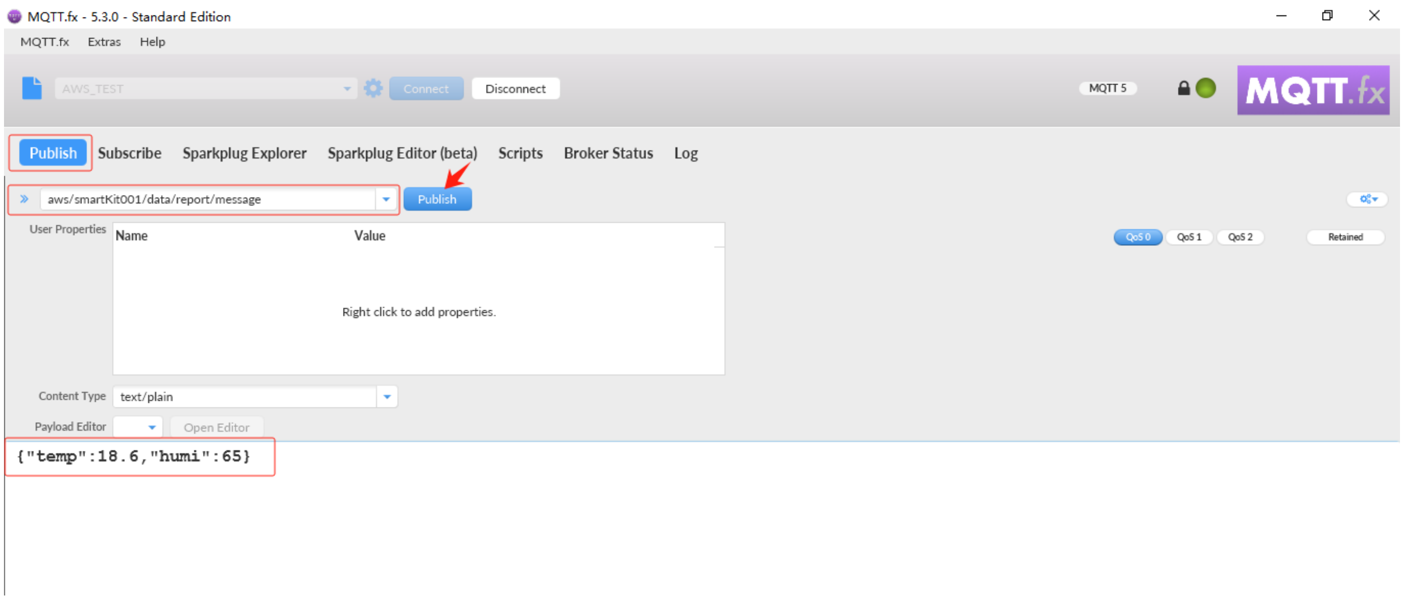
You can use the MQTT test client to monitor the MQTT messages being passed in your AWS account. Devices publish MQTT messages that are identified by topics to communicate their state to AWS IoT. AWS IoT also publishes MQTT messages to inform devices and apps of changes and events. You can subscribe to MQTT message topics and publish MQTT messages to topics by using the MQTT test client.

You can update the connection details by choosing Disconnect and making updates on the Establish connection to continue page.

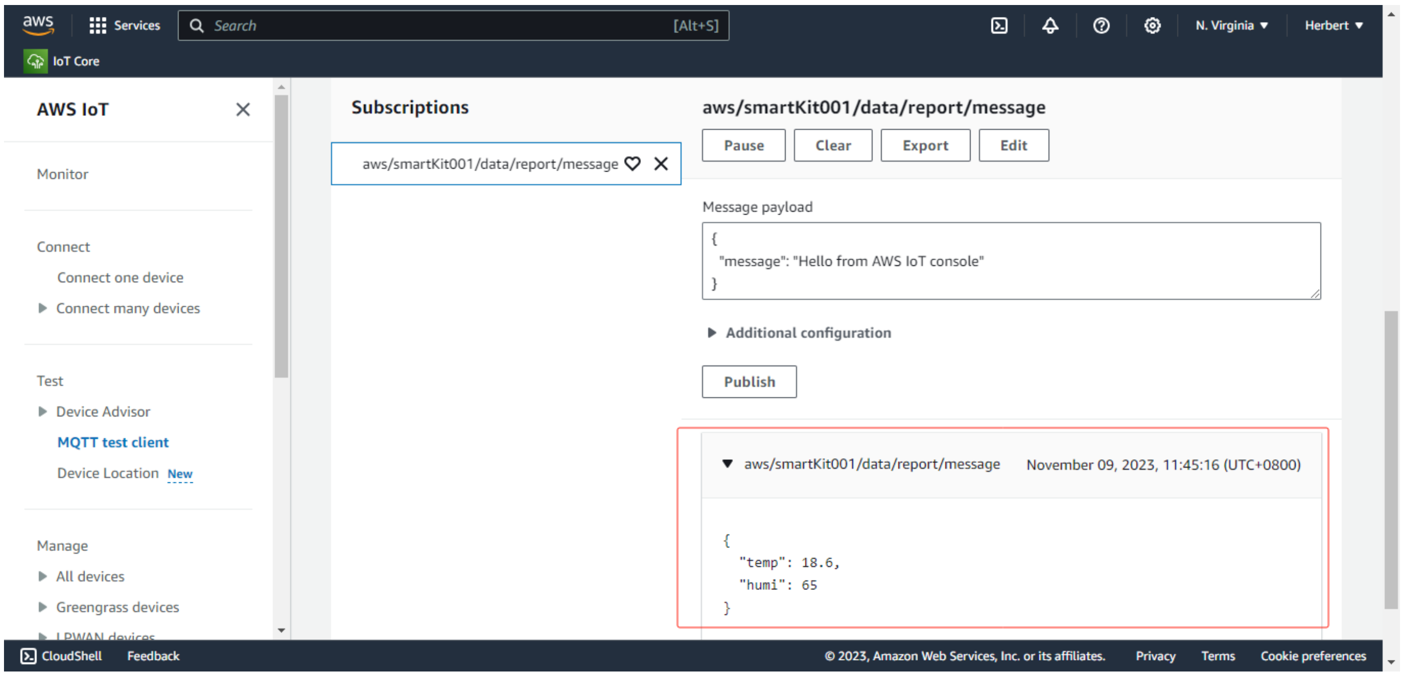
The topic filter describes the topic(s) to which you want to subscribe. The topic filter can include MQTT wildcard characters.

aws/smartKit001/data/report/message

Customize a subscribed topic
Additional configuration
Subscribe
CloudShell Feedback
© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences



```
{"temp":18.6,"humi":65}
```



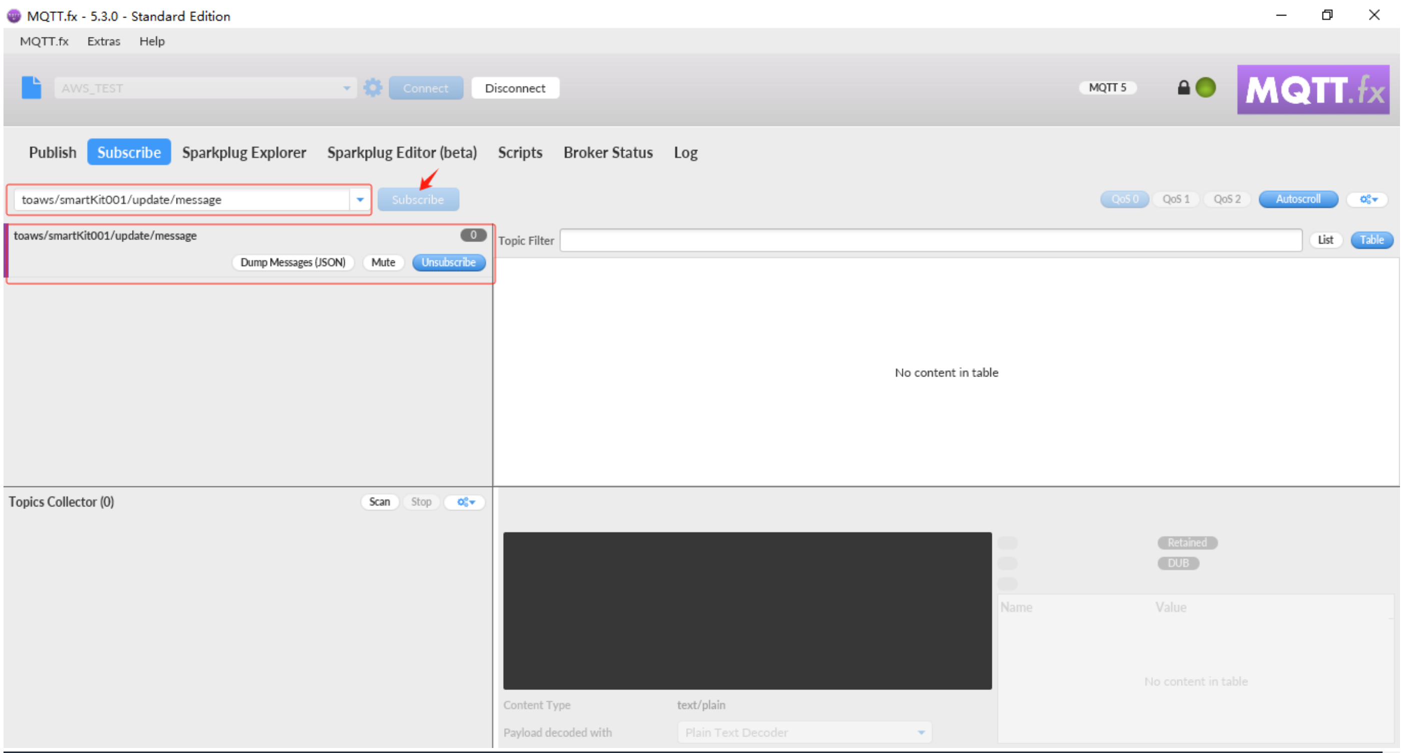
```
{
 "message": "Hello from AWS IoT console"
}
```

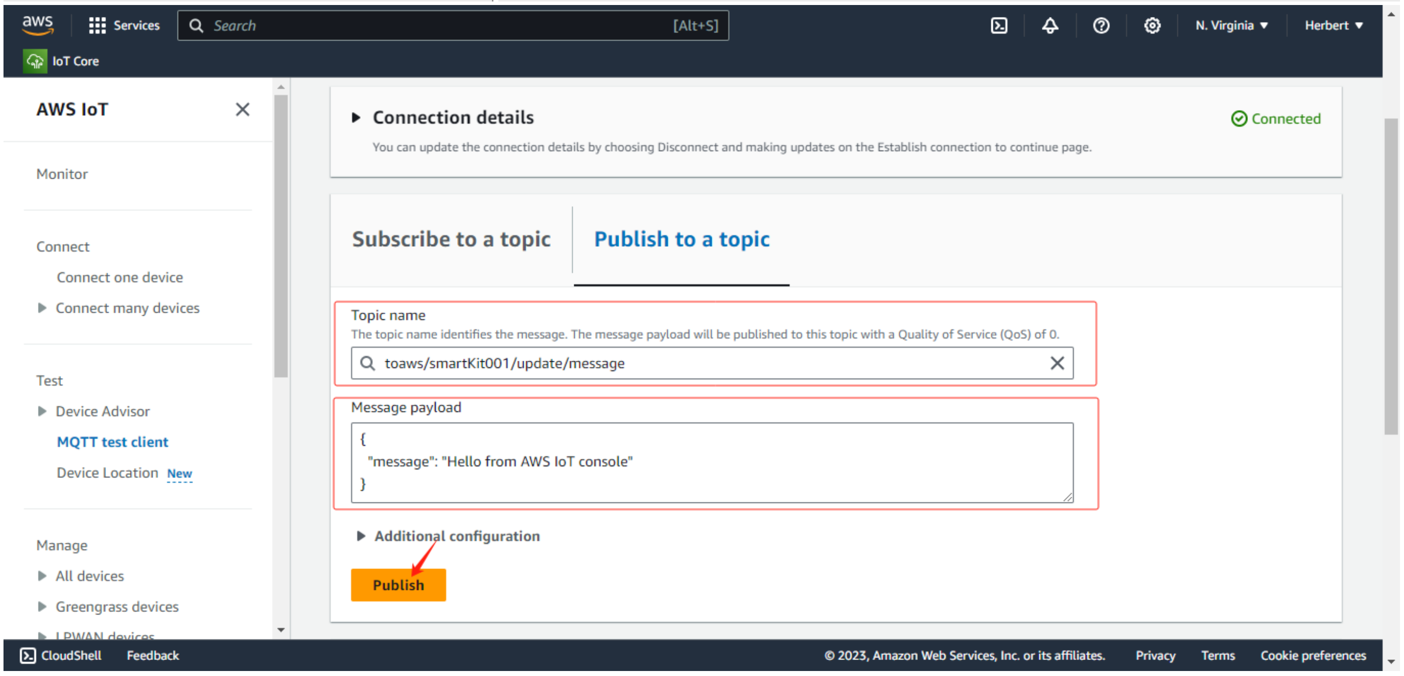
aws/smartKit001/data/report/message November 09, 2023, 11:45:16 (UTC+0800)

```
{
  "temp": 18.6,
  "humi": 65
}
```

4.3 Client Subscription

As shown below, based on the publication of "MQTT test client" to the Topic, the terminal subscribs to the defined Topic. After the test client clicks "Publish", the MQTT.fx terminal can receive the message of the subscribed Topic.

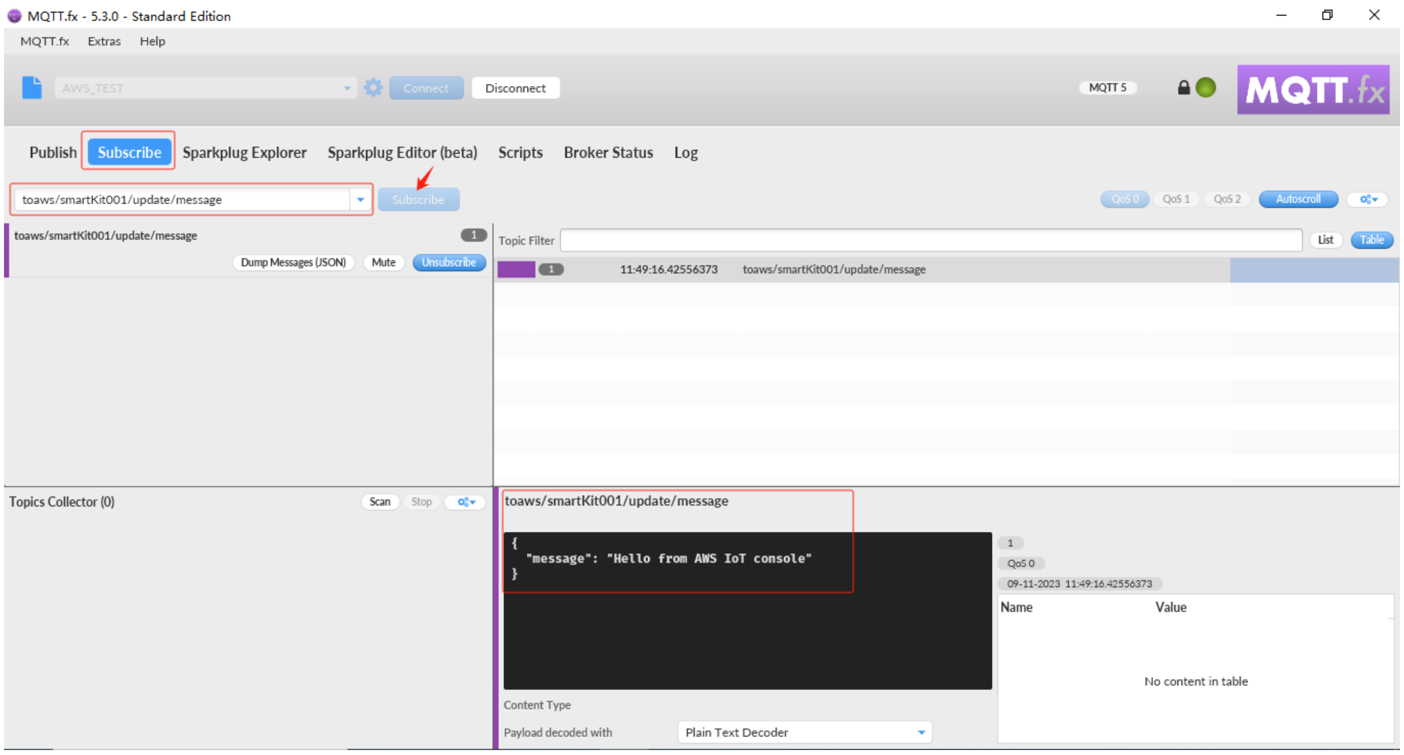




You can update the connection details by choosing Disconnect and making updates on the Establish connection to continue page.

Topic name
The topic name identifies the message. The message payload will be published to this topic with a Quality of Service (QoS) of 0.
toaws/smartKit001/update/message

Message payload
```
{
  "message": "Hello from AWS IoT console"
}
```



toaws/smartKit001/update/message

11:49:16.42556373 toaws/smartKit001/update/message

```
{
  "message": "Hello from AWS IoT console"
}
```

QoS 0
09-11-2023 11:49:16.42556373

五、Module Access to AWS IoT

5.1 Example for CAT1&CAT4 Module

The following routines for connecting to AWS IoT platform based on Quectel’s CAT1&CAT4 modules are as follows:

1）Query the network status of the device and activate the PDP

```
>> AT+CEREG?                                // Check the network registration status of the terminal
>>
>> +CEREG: 0,1                              // Successful registration network
>>
>> OK
>> AT+QENG="servingcell"                    // Check the network registration status of the terminal
>>
>> +QENG: "servingcell","NOCONN","LTE","FDD",460,11,690843E,314,1850,3,5,5,DF5C,-94,-11,-62,6,34
>>
>> OK
>> AT+QIACT=1                               // Activate PDP
>>
>> OK
>> AT+CGPADDR=1                             // Query the obtained IP address
>>
>> +CGPADDR: 1,"100.69.28.0,36.14.4.91.4.152.81.115.0.0.0.0.0.0.0.1"
>>
>> OK
```

2）Testing AWS IoT connectivity

```
>> AT+QPING=1,"a9ohm2zbim3d5-ats.iot.us-east-1.amazonaws.com"      // PING the endpoint to verify the connectivity
>>
```

```
>> OK
>>
>> +QPING: 0,"54.172.40.244",32,233,255
>>
>> +QPING: 0,"54.172.40.244",32,234,255
>>
>> +QPING: 0,"54.172.40.244",32,236,255
>>
>> +QPING: 0,"54.172.40.244",32,243,255
>>
>> +QPING: 0,4,4,0,233,243,236
```

3) Load the CA certificate and key files

```
>> AT+QFLST="RAM:*"                          // Check whether a certificate or key file has been stored in RAM
>>
>> OK
>> AT+QFUPL="RAM:cacert.pem",1187,10         // Upload the RootCA.pem to the RAM
>>
>> CONNECT
>> +QFUPL: 1187,2d19                          // The file size must be the same as the certificate size
>>
>> OK
>> AT+QFUPL="RAM:client.pem",1220,10          // Upload the certificate.pem.crt to RAM
>>
>> CONNECT
>> +QFUPL: 1220,2a                            // The file size must be the same as the certificate size
>>
>> OK
>> AT+QFUPL="RAM:user_key.pem",1679,10        // Upload the private.pem.key to RAM
>>
>> CONNECT
>> +QFUPL: 1679,6568                          // The file size must be the same as the certificate size
>>
>> OK
>> AT+QFLST="RAM:*"                           // Check the certificate file and size in RAM
>>
>> +QFLST: "RAM:cacert.pem",1187
>> +QFLST: "RAM:client.pem",1220
>> +QFLST: "RAM:user_key.pem",1679
>>
>> OK
```

4) MQTT and SSL configuration

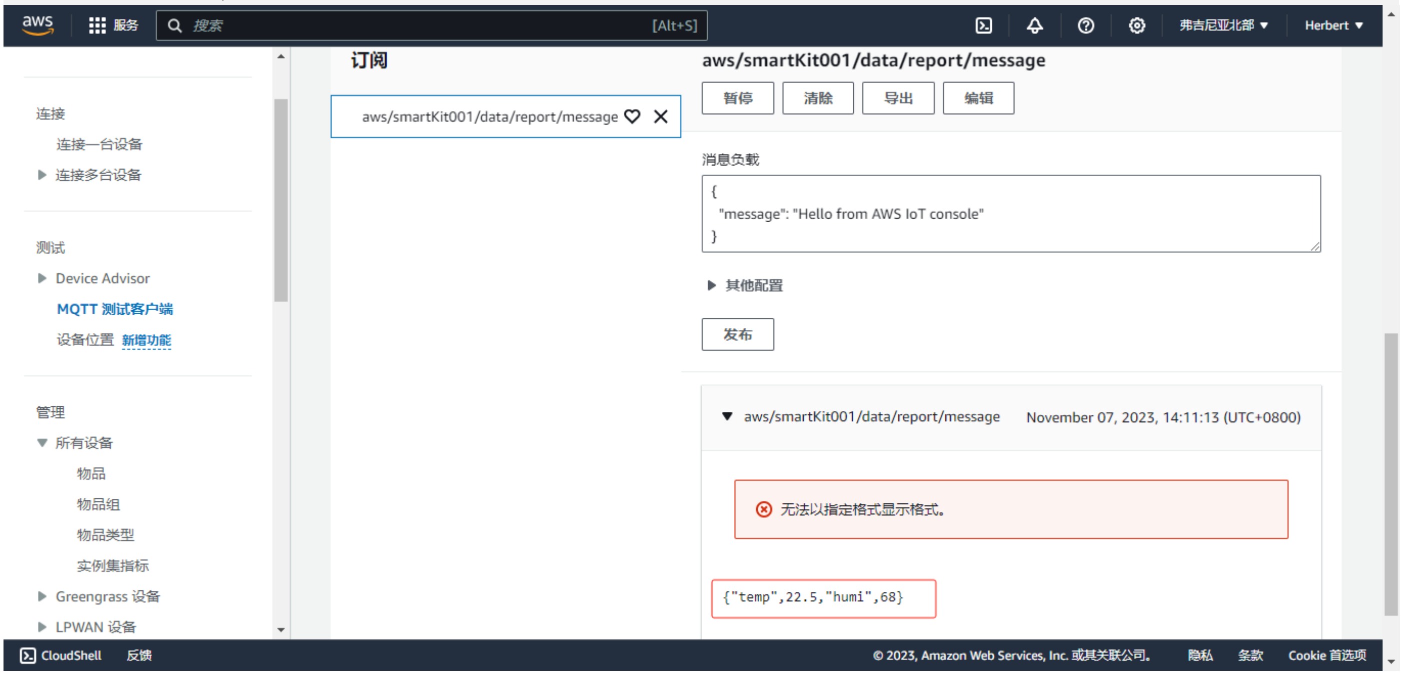
```
>> AT+QMTCFG="recv/mode",0,0,1                        // Configure receiving mode
>>
>> OK
>> AT+QMTCFG="ssl",0,1,2                              // Configure MQTT connections in SSL mode
>>
>> OK
>> AT+QSSLCFG="cacert",2,"RAM:cacert.pem"             // Configuring the CA Certificate
```

```
>>
>> OK
>> AT+QSSLCFG="clientcert",2,"RAM:client.pem"              // Configure the clientcert certificate
>>
>> OK
>> AT+QSSLCFG="clientkey",2,"RAM:user_key.pem"             // Configure the clientkey certificate
>>
>> OK
>> AT+QSSLCFG="seclevel",2,2                                // SSL authorization mode:server authentication
>>
>> OK
>> AT+QSSLCFG="sslversion",2,4                              // SSL authorized version
>>
>> OK
>> AT+QSSLCFG="ciphersuite",2,0xFFFF                        // SSL cipher suite
>>
>> OK
>> AT+QSSLCFG="ignorelocaltime",2,1                         // Ignore authorization time
>>
>> OK
```

5） MQTT of AWS IoT connect,subscribe and publish

```
>> AT+QMTOPEN=0,"a9ohm2zbim3d5-ats.iot.us-east-1.amazonaws.com",8883    // Open the MQTT SSL connection
>>
>> OK
>>
>> +QMTOPEN: 0,0
>> AT+QMTCONN=0,"smartKit001"                                 // Initiate the MQTT server connection
>>
>> OK
>>
>> +QMTCONN: 0,0,0
>> AT+QMTSUB=0,1,"toaws/smartKit001/update/message",1         // Subscribe to related topic
>>
>> OK
>>
>> +QMTSUB: 0,1,0,1
>> AT+QMTPUBEX=0,1,1,0,"aws/smartKit001/data/report/message",23    // Publish messages to related topic
>>
>> > {"temp",22.5,"humi",68}
>>
>> OK
>>
>> +QMTPUBEX: 0,1,0
>>
>> +QMTRECV: 0,0,"toaws/smartKit001/update/message",42,"{ "message": "Hello from AWS IoT console"}"
>> AT+QMTCLOSE=0                                              // Initiating MQTT connection disconnected
>>
>> OK
```

```
>>
>> +QMTCLOSE: 0,0
```



aws/smartKit001/data/report/message

消息负载

```
{
  "message": "Hello from AWS IoT console"
}
```

aws/smartKit001/data/report/message   November 07, 2023, 14:11:13 (UTC+0800)

无法以指定格式显示格式。

```
{"temp",22.5,"humi",68}
```

© 2023, Amazon Web Services, Inc. 或其关联公司。

5.2 Example for NB-IoT Module(BC660K)

The following routines for connecting to AWS IoT platform based on Quectel’s NB-IoT modules are as follows:

1） Query the network status of the device and IP

```
>> AT+CEREG?                                  // Check the network registration status of the terminal
>>
>> +CEREG: 0,1                                 // Successful registration network
>>
>> OK
>> AT+CGPADDR?                                 // Query the obtained IP address
>>
>>+CGPADDR: 0,"10.5.29.72","2409:8d30:0104:02e2:17b5:74f2:a402:e8be"
>>
>> OK
```

2） Testing AWS IoT connectivity

```
>> AT+QIDNSCFG=0,"8.8.8.8","8.8.4.4"
>>
>> OK
>> AT+QPING=0,"a9ohm2zbim3d5-ats.iot.us-east-1.amazonaws.com"     // PING the endpoint to verify the connectivity
>>
>> OK
>>
>> +QPING: 0,54.163.74.32,32,336,237
>>
>> +QPING: 0,54.163.74.32,32,354,237
>>
>> +QPING: 0,54.163.74.32,32,394,237
>>
```

```
>> +QPING: 0,54.163.74.32,32,431,237
>>
>> +QPING: 0,4,4,0,336,431,378
```

3） Load the CA certificate and key files

```
>> AT+QSSLCFG=0,0,"cacert"                       // Upload the RootCA.pem to the FILE system
>>
>> >
>> -----BEGIN CERTIFICATE-----
>> MIIDQTCCAimgAwIBAgITBmyfz5m/jAo54vB4ikPmljZbyjANBgkqhkiG9w0BAQsF
>> ......
>> rqXRfboQnoZsG4q5WTP468SQvvG5
>> -----END CERTIFICATE-----
>> +QSSLCFG: 0,0,"cacert",1187
// The byte size of the indication must correspond to the byte size of the content of AmazonRootCA1.pem
>>
>> OK
>> AT+QSSLCFG=0,0,"clientcert"                   // Upload the certificate.pem.crt to the FILE system
>>
>> >
>> -----BEGIN CERTIFICATE-----
>> MIIDWjCCAkKgAwIBAgIVAKp/znqyrMfoSWNqhC/Ln+qsPPI4MA0GCSqGSIb3DQEB
>> ......
>> ++kMnCN/oHJzDl2jzL65XktQm99MojobbN121jZm3v34nqOuYFT6351CaA64DQ==
>> -----END CERTIFICATE-----
>>
>> +QSSLCFG: 0,0,"clientcert",1224
// The byte size of the indication must correspond to the byte size of the content of xxx-certificate.pem.crt
>>
>> OK
>> AT+QSSLCFG=0,0,"clientkey"                    // Upload the private.pem.key to the FILE system
>>
>> >
>> -----BEGIN RSA PRIVATE KEY-----
>> MIIEowIBAAKCAQEA3rClmNdAS6x43he0Zn0hq7BGHrdt3ttIjYmTsCVttkcdGX3+
>> ......
>> cnEBCR2U3DpU1qNrn0D8r8qCJjWbYJwzEbEoFaKi8zHuhLLVuuus
>> -----END RSA PRIVATE KEY-----
>>
>> +QSSLCFG: 0,0,"clientkey",1675
// The byte size of the indication must correspond to the byte size of the content of xxx-private.pem.key
>>
>> OK
```

4） MQTT and SSL configuration

```
>> AT+QSSLCFG=0,0,"seclevel",2                   // SSL authorization mode:server authentication
>>
>> OK
>> AT+QMTCFG="ssl",0,1,0,0                       //Enable SSL and configure SSL context/connect index
>>
```

```
>> OK
>> AT+QMTCFG="version",0,1                          //Configure the MQTT version
>>
>> OK
>> AT+QSSLCFG=0,0                                   //Query and validate the results of the current configuration
>>
>> +QSSLCFG: 0,0,"seclevel",2
>> +QSSLCFG: 0,0,"sslversion",4
>> +QSSLCFG: 0,0,"dataformat",0,0
>> +QSSLCFG: 0,0,"timeout",90
>> +QSSLCFG: 0,0,"debug",0
>> +QSSLCFG: 0,0,"cacert",1187
>> +QSSLCFG: 0,0,"clientcert",1224
>> +QSSLCFG: 0,0,"clientkey",1675
>> +QSSLCFG: 0,0,"dtls",0
>> +QSSLCFG: 0,0,"dtlsversion",2
>>
>> OK
```

5） MQTT of AWS IoT connect,subscribe and publish

```
>> AT+QMTOPEN=0,"a9ohm2zbim3d5-ats.iot.us-east-1.amazonaws.com",8883   // Open the MQTT SSL connection
>>
>> OK
>>
>> +QMTOPEN: 0,0
>> AT+QMTCONN=0,"smartKit001"                                            // Initiate the MQTT server connection
>>
>> OK
>>
>> +QMTCONN: 0,0,0
>> AT+QMTSUB=0,1,"toaws/smartKit001/update/message",1                    // Subscribe to related topic
>>
>> OK
>>
>> +QMTSUB: 0,1,0,1
>> AT+QMTPUB=0,1,1,0,"aws/meter001/data/report/message",23,"{"temp",22.5,"humi",68}"
// Publish messages to related topic
>>
>> OK
>>
>> +QMTPUB: 0,1,0
>>
>> +QMTRECV: 0,0,"toaws/smartKit001/update/message","{ "message": "Hello from AWS IoT console"}"
>> AT+QMTCLOSE=0                                                      // Initiating MQTT connection disconnected
>>
>> OK
>>
>> +QMTCLOSE: 0,0
```

六、Troubleshooting Abnormal Issues

If your device fails to connect to the AWS IoT platform, please refer to the following troubleshooting advice:

1) For details, see the procedure and methods in the document.
2) Check whether your device or module is successfully connected to the radio network.
3) Check whether your device or module successfully activates PDP and obtains IP address.
4) Please first verify whether the device data endpoint PING is successful.
5) Check whether the security policy, including TLS version and cipher suite, matches the module configuration parameters.
6) Check whether your CA certificate and key files are completely loaded into FILE.