

Lab: Add Voice Control Feature to AWS IoT Solutions

Overview

AWS IoT Device Shadow provides the ability to connect devices with your apps so that you can control the devices in your app. And Alexa Voice Service (AVS) enables you to add voice-powered experiences to your connected devices.

In this lab, you will learn how to add voice control features to your AWS IoT Solutions.



AWS IoT

Objectives

After completing this lab, you will be able to:

- Create a custom Alexa Skill Kit
- Create a Lambda function to integrate with AWS IoT Service Device Shadow

Pre-requisites	<p>This lab requires:</p> <ul style="list-style-type: none">• AWS account• Access to a notebook computer with Wi-Fi running Microsoft Windows, Mac OS X, or Linux (Ubuntu, SuSE, or Red Hat).• For Microsoft Windows users: Administrator access to the computer.• An Internet browser such as Chrome, Firefox. With camera support would be prefer.
Duration	40 to 60 Minutes

v.1.0.2.0705

Task 1: Create a device in AWS IoT Service

Overview

To connect a device to AWS IoT, we recommend you first create a device in the thing registry. This registry allows you to keep a record of all of the devices that are connected to your AWS IoT account.


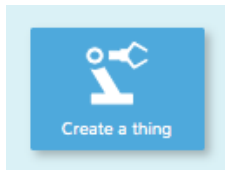


AWS IoT

Task 1-1: Create a Device in the Thing Registry

Overview

In this task you will create a Device in the Thing Registry for further usage in this lab..

Step	Instruction
1.1.1	<p>Sign in AWS Console, make sure you are using Asia Pacific (Tokyo) region, and the select AWS IoT from the menu:</p> 
1.1.2	<p>On the Resources page, select + Create a resource button, and then click on the Create a thing item:</p> 

1.1.3	Enter “ myLight ” in the name column, and click Create button。
1.1.4	Now click View thing button and you can see the device related information on the right-side panel including REST API end poinr and MQTT Topic....etc.
1.1.5	Click on Update Shadow in the right panel.
1.1.6	Update the Shadow state as below: <pre>{ "desired": {}, "reported": {"light":0} }</pre>
1.1.7	Click on the Update shadow button, and you will see the new Shadow state and Shadow metadata in the Details page:
1.1.8	On the Resources page, select x Close create panel button. Finish create a device in the thing registry.


Task 1-2: Creating and Activate an AWS IoT Device Certificate and Policy

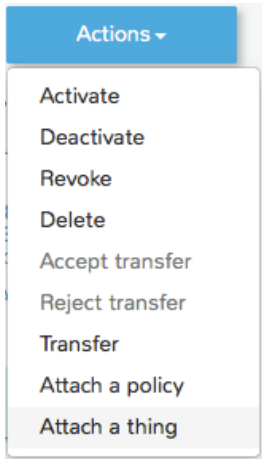
Overview


Communication between your AWS IoT button and AWS IoT is protected through the use of X.509 certificates. AWS IoT can generate a certificate for you or you can use your own X.509 certificate.

AWS IoT policies are used to authorize your button to perform AWS IoT operations, such as subscribing or publishing to MQTT topics. Your button will present its certificate when sending messages to AWS IoT. To allow your button to perform AWS IoT operations, you must create an AWS IoT policy and attach it to your device certificate.

In this task we will use AWS IoT to generate an X.509 certificate and activate it for you. Also we will create an AWS IoT policy and attach it to the device certificate.

Step	Instruction
1.2.1	<p>On the Resources page, select + Create a resource → Create a certificate :</p> 

1.2.2	<p>Click 1-Click certificate create button, and AWS IoT will create the keys and certificate for you:</p> <div style="background-color: #e1f5fe; padding: 10px; margin: 10px 0;"> <p>Your new certificate has been created. You can attach a certificate to a thing so it can connect to AWS IoT and attach a policy to give it permissions.</p> <p>Please download these files and save them in a safe place. Certificates can be retrieved at any time, but the private and public keys will not be retrievable after closing this form.</p> <ul style="list-style-type: none"> • Show public key • Show private key • Show certificate </div> <p>[Note] Please DO NOT close this page before finish downloading the keys and certificate.</p>
1.2.3	<p>Create an IoT Certs folder in your filesystem. On the Resources page, choose the Download private key and Download certificate links, and save the private key and certificate to your computer.</p>
1.2.4	<p>On the Resources page, select x Close create panel button.</p>
1.2.5	<p>Select the check box on the certificate which we just created, and from the Actions menu, choose Attach a thing.</p> <div style="margin: 10px 0;">  </div>
1.2.6	<p>In the Confirm windows, type myLight in the Thing name column, and then select Attach button.</p>
1.2.7	<p>Repeat 1.2.5, select Actions, and then select Activate from the menu. AWS IoT now can use this certificate to identify this device.</p>

1.2.8	<p>On the Resources page, select + Create a resource → Create a policy :</p> 
1.2.9	<p>In the Create a policy section, type myFirstThing-policy in the Name column. From the Action menu, choose iot:*. In the Resource field, type * to allow access to all AWS IoT resources. Select the Allow check box, choose Add statement, and then choose Create.</p>
1.2.10	<p>Now we need to attach this policy to the device certificate. Repeat 1.2.5, select Actions, and then select Attach a policy from the menu.</p>
1.2.11	<p>In the Confirm windows, type myFirstThing-policy in the Policy name column, and then select Attach button.</p>

Task 2: Create an Alexa Skill Kit

Overview

AWS IoT Service can integrate with your applications to let users to manage their devices. In this task, you will learn how to create an Alexa Skill Kit to control your device via AWS IoT.

Alexa Voice Service and AWS IoT



Task 2-1: Create a Lambda function for Alexa Skill Kit

Overview

The easiest way to build the cloud-based service for a custom Alexa skill is by using AWS Lambda, so there is no need to provision or continuously run servers. AWS Lambda supports JavaScript and you upload the code for your Alexa skill to a Lambda function and Lambda does the rest, executing it in response to Alexa voice interactions and automatically managing the compute resources for you.

Step	Instruction
2.1.1	Open AWS Lambda Console, and click on Create a Lambda function → Skip .

2.1.2	In the Configure function page, named the function skill-light-control , and add some description. Make sure the Runtime is Node and Edit code inline is selected as well.
2.1.3	Please download the Alexa Skill Lambda template from: http://bit.ly/Alexa-Mini-Lambda . Open the file downloaded and copy/paste the contents of the file to Lambda function code.
2.1.4	<p>Add below code at the beginning of the Lambda code:</p> <pre>//Environment Configuration var config = {}; config.IOT_BROKER_ENDPOINT = "[AWS IoT endpoint of your thing].toLowerCase(); config.IOT_BROKER_REGION = "us-east-1"; config.IOT_THING_NAME = "myLight"; var AWS = require('aws-sdk'); AWS.config.region = config.IOT_BROKER_REGION; var iotData = new AWS.IotData({endpoint: config.IOT_BROKER_ENDPOINT});</pre>

2.1.5

Add below code to the end of the Lambda code:



```
function lightUp (intent, session, callback) {  
  
  var repromptText = null;  
  
  var sessionAttributes = {};  
  
  var shouldEndSession = true;  
  
  var speechOutput = "";  
  
  
  var payloadObj={ "state":  
    { "desired":  
      { "light":1}  
    }  
  };  
  
  
  //Prepare the parameters of the update call  
  
  var paramsUpdate = {  
    "thingName" : config.IOT_THING_NAME,  
    "payload" : JSON.stringify(payloadObj)  
  };  
  
  
  //Update Device Shadow  
  
  iotData.updateThingShadow(paramsUpdate, function(err, data) {  
    if (err){  
      //Handle the error here  
    }
```

	<pre> else { speechOutput = "The light has been turned on!"; console.log(data); callback(sessionAttributes,buildSpeechletResponse(intent.name, speechOutput, repromptText, shouldEndSession)); } }); } </pre>
2.1.6	<p>Find the onIntent method, modified the code as below:</p> <pre> function onIntent(intentRequest, session, callback) { console.log("onIntent requestId=" + intentRequest.requestId + ", sessionId=" + session.sessionId); var intent = intentRequest.intent, intentName = intentRequest.intent.name; lightUp(intent,session,callback) } </pre>
2.1.7	<p>In the Lambda function handler and role, select Create new role → *Basic execution role in the Role column.</p> <p>In the new prompt window, make sure the new role have the iot DataAccess policy.</p>
2.1.8	<p>In the new Windows, select Allow.</p>
2.1.9	<p>In the Lambda console, select Next → Create function.</p>
2.1.10	<p>After the Lambda function created, select the Event sources tab.</p>

2.1.11	Select Add event source , and select Alexa Skills Kit in the Event source type. Then click on the Submit button.
2.1.12	Copy the ARN of the Lambda function on the right-top corner. The ARN format should be like this: arn:aws:lambda:us-east-1:XXXXXXXXXX:function:skill-light-control

Task 2-2: Create an Alexa Skills Kit

Overview	In this task, we will create a new Alexa Custom Skills Kit to trigger the Lambda function
-----------------	---

Step	Instruction
2.2.1	Sign up and login Amazon Developer Portal (https://developer.amazon.com) with your Amazon account.
2.2.2	Click on Alexa , and then click on Get Started of Alexa Skill Kit. <div> <p>Get started with Alexa</p> <p>Add new voice-enabled capabilities using the Alexa Skills Kit, or add voice-powered experiences to your connected devices with the Alexa Voice Service.</p> <div> <div>  <p>Alexa Skills Kit Easily add new skills to Alexa Get Started ></p> </div> <div>  <p>Alexa Voice Service Bring voice capabilities to your connected device Get Started ></p> </div> </div> </div>
2.2.3	Select Add a New Skill.
2.2.4	In the Create a New Alexa Skill page, select Custom Interaction Model as the skill type. Enter Light Control in the name field and my agent in the invocation name field. Click Next when finished.

2.2.5	<p>In the Interaction Model page, enter below script into Intent Schema:</p> <pre>{ "intents": [{ "intent": "LightIntent" }] }</pre>
2.2.6	<p>Enter below script into Sample Utterances and then click Next:</p> <p>LightIntent turn on the light.</p>
2.2.7	<p>In the Configuration page, select Lambda ARN as the endpoint and paste the ARN which you copied in the step 2.1.12. It should be similar as below:</p> <p>arn:aws:lambda:us-east-1:xxxxxxxxxx:function: skill-light-control</p>
2.2.8	<p>Select No on the account linking, and then click Next.</p>

2.2.9

Now we can test the voice control feature in the Test page. In the Service Simulator section, enter **“Alexa, ask my agent to turn on the light.”** Into the Enter Utterance field and click on the **Ask Light Control** button. Then you will find “The light has been turned on!” output speech message in the Lambda Response as below:

Lambda Response


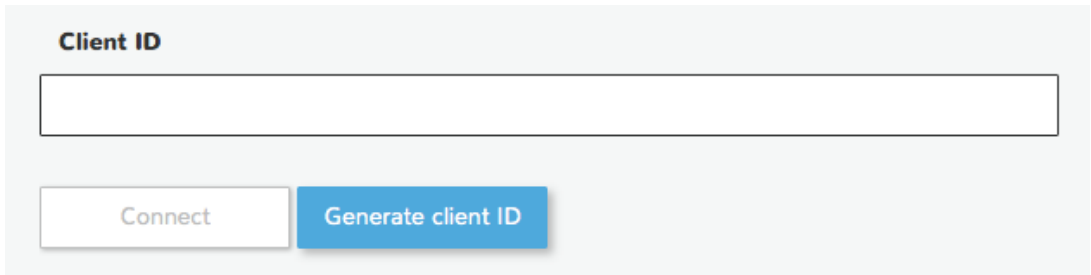
```
1 {  
2   "version": "1.0",  
3   "response": {  
4     "outputSpeech": {  
5       "type": "PlainText",  
6       "text": "The light has been turned on!"  
7     },  
8     "card": {  
9       "content": "SessionSpeechlet - The light ha  
10      "title": "SessionSpeechlet - LightIntent",  
11      "type": "Simple"  
12    },  
13  }
```

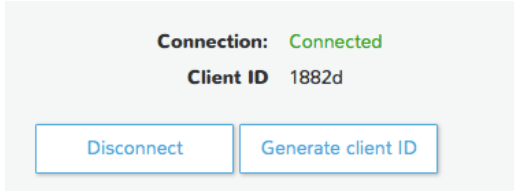
Task 2-3: View Device MQTT Messages with the AWS IoT MQTT Client

Overview

Before connected to a real device, you can use AWS IoT MQTT client to better understand the MQTT messages sent by a device.

Devices publish MQTT messages on topics. You can use the AWS IoT MQTT client to subscribe to these topics to see the content of these messages.

Step	Instruction
2.3.1.	In the AWS IoT console, choose MQTT Client .
2.3.2.	<p>Select Device Gateway connection:</p> 
2.3.3.	<p>Choose Generate client ID to create a client id which will be included in each AWS IoT message:</p> 

<p>2.3.4.</p>	<p>Click on Connect button and connect to AWS IoT service. The Connect status will be "Connected" if success:</p> 
<p>2.3.5.</p>	<p>Then subscribe to the topic on which your thing publishes. Choose Subscribe to topic, and in Subscription topic, type \$aws/things/myLight/shadow/update/delta, and then choose Subscribe.</p>
<p>2.3.6.</p>	<p>Execute step 2.2.9 again, then MOTT Client will receive the delta message as below:</p> 