ON Semiconductor

Is Now



To learn more about onsemi™, please visit our website at www.onsemi.com

onsemi and ONSEMI. and other names, marks, and brands are registered and/or common law trademarks of Semiconductor Components Industries, LLC dba "onsemi" or its affiliates and/or subsidiaries in the United States and/or other countries. onsemi owns the rights to a number of patents, trademarks, copyrights, trade secrets, and other intellectual property. A listing of onsemi product/patent coverage may be accessed at www.onsemi.com/site/pdf/Patent-Marking.pdf. onsemi reserves the right to make changes at any time to any products or information herein is provided "as-is" and onsemi makes no warranty, representation or guarantee regarding the accuracy of the information, product features, availability, functionality, or suitability of its products for any particular purpose, nor does onsemi assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation special, consequential or incidental damages. Buyer is responsible for its products and applications using onsemi products, including compliance with all laws, regulations and safety requirements or standards, regardless of any support or applications information provided by onsemi. "Typical" parameters which may be provided in onsemi data sheets and/ or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. onsemi does not convey any license under any of its intellectual property rights nor the rights of others. onsemi products are not designed, intended, or authorized for use as a critical component in life support systems or any FDA Class 3 medical devices or medical devices with a same or similar classification in a foreign jurisdiction or any devices intended for implantation in the human body. Should Buyer purchase or use onsemi products for any such unintended or unauthorized application,

Connecting the IDK to the AWS Cloud

Introduction

This application note provides details on connecting the IoT Development Kit (IDK) to the AWS Cloud. An environmental sensing (temperature, humidity, pressure, ambient light) use case provided as an example within the IDK Software will be used to demonstrate connectivity to AWS.

Sensor data are transmitted over secure connection to AWS IoT Core service. Connection is secured by TLS encryption and the measured sensor data are periodically sent in JSON format over MQTT.

References

[Ref 1] IoT IDK User Guide:

https://www.onsemi.com/pub/Collateral/AND9666-D.PDF

[Ref 2] AWS IoT Developer Guide:

https://docs.aws.amazon.com/iot/latest/developerguide/iot-dg.pdf

Prerequisities

The below items are needed in order to successfully configure and run this use case.

Hardware

- IDK Baseboard (BB-GEVK) with WizFi250 Wi-Fi module
- Multi-Sensor shield (MULTI-SENSE-GEVB)

The Multi-Sensor shield should be plugged into one of the two available PMOD connectors of the IDK Baseboard. The IDK Baseboard should then be connected to PC to provide power and for downloading the example use case.



Figure 1.



ON Semiconductor®

www.onsemi.com

APPLICATION NOTE

Software

• IoT IDK Software (version 4.3 & above) : <u>BB-GEVK</u> <u>Software</u>

Refer to IoT IDK User Guide (Ref 1) for installation details.

AWS Account

AWS account with IoT Core service created
 Please follow instructions at the below link to create an
 AWS account and setup an IoT Core instance:

<u>https://docs.aws.amazon.com/iot/latest/developerguide/iot-console-signin.html</u>

Setup

Once the hardware, software and an AWS account are setup, there are two main steps involved in exercising the node-to-cloud pathway.

- 1. AWS IoT Core Configuration
- 2. IDK Use Case Modification, Compilation and Download

AWS IoT Core Configuration

This section lists steps needed to configure the AWS IoT Core to interface with the hardware. All of these steps can be found in the <u>AWS IoT Developer Guide</u> as well.

Creating Security Policy for IDK Baseboards

A security policy defines permissions for all connected IoT devices (Things). Policies can be used to restrict devices' ability to read/write data to the cloud by creating sets of rules that are attached to each IoT Thing. Ideally there should be a custom policy for each connected device (device group) but they can also be shared by multiple IoT devices.

To create a policy for this exercise, navigate to the Secure -> Policies subpage in the AWS IoT console. This page contains a list of existing policies and allows creation of new policies. Generating a new policy can be generated by clicking the Create button.

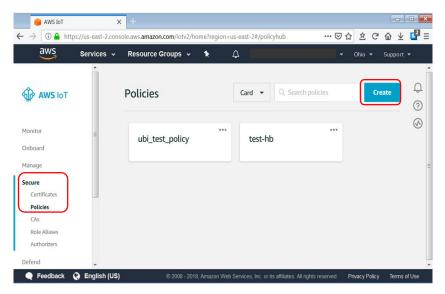


Figure 2.

 After clicking the Create button fill out the policy name (ms_sensor_node_policy for this example).
 In the Add statements sections click on Advanced mode and paste the policy shown below. After that click Create button to confirm and create new policy.

Figure 3.

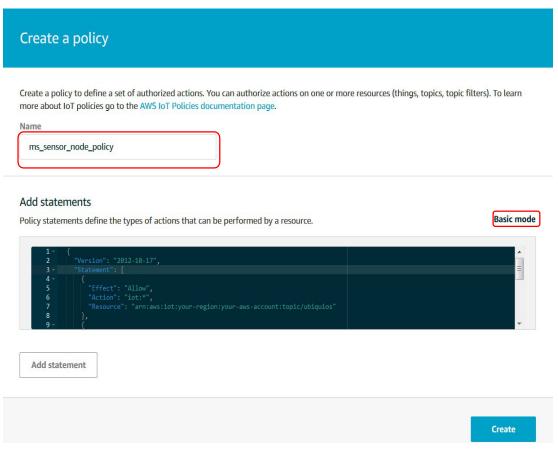


Figure 4.

Create IoT Thing

Things are way to represent devices in the AWS IoT Core platform. A "Thing" needs to be created for each device that will be connected to the cloud.

To create a thing navigate to the Manage ->
 Things subpage of IoT Core console. This page
 contains list of all created Things and allows to
 create new Things by clicking the Create button.

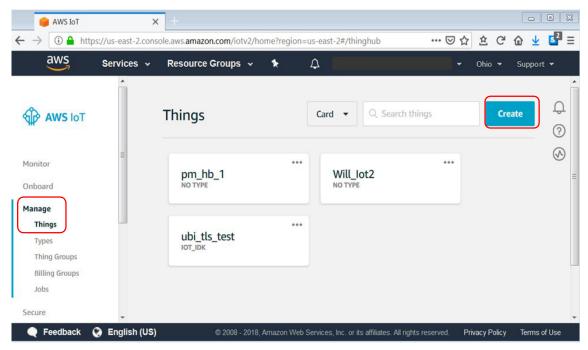


Figure 5.

2. On the next page select: Create a single thing

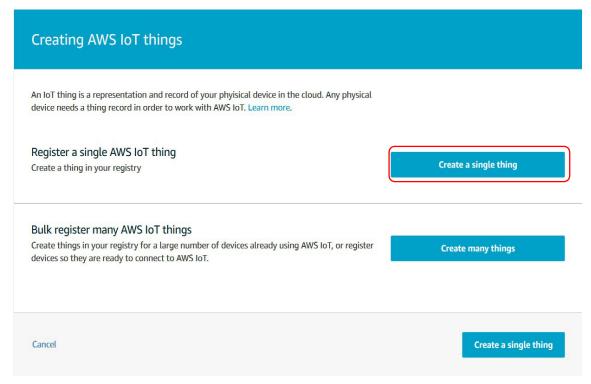


Figure 6.

- 3. On the next page (step 1/3) enter name of the new Thing. ms_sensor_board will be used in this example.
 - Optionally select a Thing Type for this device.
- 4. In step 2 use the **One-click certificate creation** option by clicking the **Create Certificate** button.
- 5. On next page first click Activate button.
- 6. When the certificate activates **download all three generated certificate files to PC**.

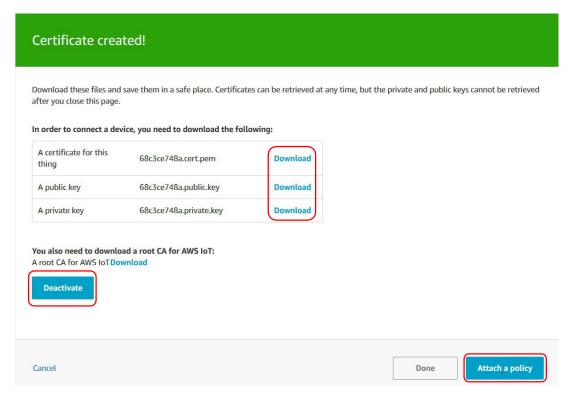


Figure 7.

7. Once all three files are downloaded click the **Attach a policy** button to attach the previously created policy to this Thing.

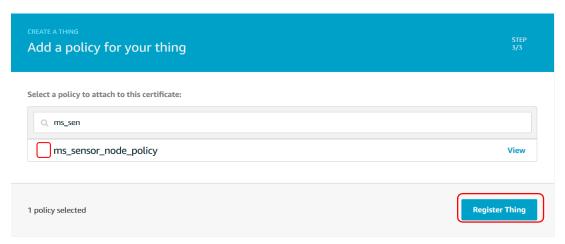


Figure 8.

- 8. Finally click the **Register Thing** button to finish Thing creation process. After this step the browser will redirect back to the list of all created things and there should be a new thing called **ms sensor board**.
- 9. More details about the Thing will be displayed by clicking on the device in the list.

The AWS endpoint needs to be copied from the *Interact* subpage of Thing details page as shown below.

(This will be required later when configuring the program that will be downloaded to the IDK Baseboard)

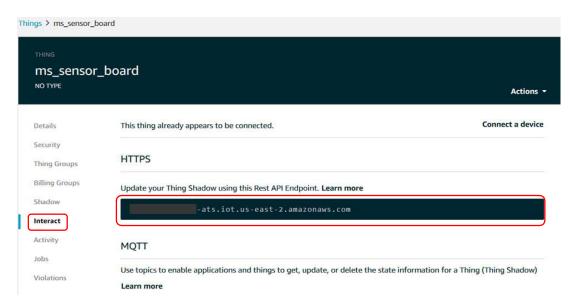


Figure 9.

Configuring the IDK Example Project

This section walks you through all the steps involved in entering the credentials of the "Thing" in the example use case. Please refer to the IoT IDK User Guide (Ref 1) for installation and launch of the example use case.

Project File Structure

The structure of the project is as follows:

- src Folder containing all source code files.
 - configuration.h Contains macro definitions for Wi-Fi credentials and cloud related variables that need to be modified (steps in next section)
 - aws_thing_keystore.def Contains preprocessed device certificates that are used to authenticate the device when it connects to cloud. This file needs to be generated and replaced for each device. (steps in next section)
 - aws_iot.c Main application file that handler Wi-Fi connectivity, AWS IoT session and timers for periodic publishing of sensor data to cloud.
 - sensor_node.* Abstraction layer for the IDK sensor libraries used in this example
 - platform.* Part of the TLS stack (from <u>Ubiquios</u>).
 Should not be modified.
 - pmalloc_pools.* Part of the TLS stack (from <u>Ubiquios</u>). Should not be modified.
 - hal Folder containing abstraction layers for the Ubiquios TLS stack. Should not be modified.

 aws_ats_ca_callback.def - Data used for verification of Amazon IoT Core server certificate chain. Should not be modified.

Entering Credentials to configuration.h file

The following macros in this file need to be modified to match your settings.

 Entering correct Wi-Fi SSID and passphrase for Wi-Fi network that will be used In this example the network is called *MyTestWifi* and uses the passphrase *mysecretpassphrase*:

```
#define AP_SSID MyTestWifi
#define AP_PASSPHRASE mysecretpassphrase
```

Figure 10.

- 2. Entering AWS related Thing variables. This includes things endpoint, topic name and client ID. Endpoint can be obtained from AWS console in the Interact subpage of Things details page. (step 9 from the "Create IoT Thing" section). Topic name depends on applied policy. If the policy from this document was used it can be any valid MQTT topic name. Client id can be chosen arbitrarily and has to be unique for each connected Thing (MQTT limitation). (See Figure 11.)
- 3. Optionally set custom send and retry intervals.

```
#define AWS_IOT_ENDPOINT alv6q8shdhn4jf-ats.iot.us-east-2.amazonaws.-
com #define AWS_IOT_TOPIC ms_node_test
#define AWS_IOT_CLIENT_ID ms_node_123456789
```

Figure 11.

Generating Key Store from Downloaded Certificates

AWS uses certificates to authenticate Things that are attempting to connect to the IoT platform. To overcome the RAM limitation of the IDK baseboard processor, the downloaded certificates cannot be used directly but have to be preprocessed and included as a source code file into the project.

The following steps need to be done to get preprocessed certificate definition file that can be used in the project:

 Copy the three downloaded certificate files (x.cert.pem, x.public.key, x.private.key) into the following folder inside of the IDK software installation directory:

C:\\IDK_INSTALL_DIR\external\ubiquios\bin

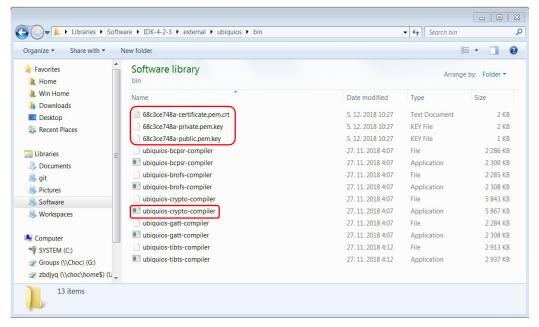


Figure 12.

2. Open **cmd.exe** terminal in this directory. This can be done by typing cmd.exe into the path section on

top of file explorer or running the cmd.exe and navigating to the folder manually.

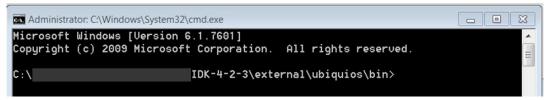


Figure 13.

3. Run the following command in the terminal replacing the 68c3ce748a with your certificates: **ubiquios-crypto-compiler.exe keystore**

<68c3ce748a>-private.pem.key <68c3ce748a>-certificate.pem.crt -n aws_thing_keystore -o aws_thing_keystore.def

```
Administrator: C:\Windows\System32\cmd.exe

Microsoft Windows [Version 6.1.7601]

Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\

IDK-4-2-3\external\ubiquios\bin>ubiquios-crypto-comp iler.exe keystore 68c3ce748a-private.pem.key 68c3ce748a-certificate.pem.crt.txt -n aws_thing_keystore -o aws_thing_keystore.def

C:\

IDK-4-2-3\external\ubiquios\bin>
```

Figure 14.

4. Copy generated **aws_thing_keystore.def** file into the IDK projects src folder. This will overwrite the default empty file that was in the project.

Verifying Demo Functionality

Once all of the above steps are done the program can be compiled and downloaded to the IDK Baseboard. If both

Wi-Fi and AWS credentials are correct the terminal output from the board should look similar to the image below:

```
C:\cygwin64\home\zbdjyg\opt\IDK-4-2-3\IDK\Flash_utility.exe
                                                                              open_serial_port : opening of serial port successful
   This version of ubiquios is for evaluation purposes only and must not be dis
ributed.
Welcome to the Internet of Thinas - powered by ubiquios!
Build version: ubiquios-7.7.4-638 20181127030525
Sensor node initialization success
Attempting to connect to network.
SSID =
passphrase =
Connecting..
Subscribing to topic 'ms_node_test'
Link came up with IP address 192.168.43.53 (active if: wlan)
AWS IoT connection succeeded
Generated sensor data payload:
 "timestamp":33227,"temp<sup>=</sup>:23.02,"hum":31.51,"pres":101138,"als":108.00}
Sending message
AWS IoT send completed
AWS IoT subscribe succeeded
Received an AWS IoT message
    Topic: ms_node_test
    Message body:
        {"timestamp":33227,"temp":23.02,"hum":31.51,"pres":101138,"als":108.00}
```

Figure 15.

- 1. Wi-Fi connection was successful and the board was assigned an IP address.
- 2. AWS IoT connection was successfully established
- 3. Sensor measurement completed and correct data can be seen in the JSON formatted payload.
- 4. The message was send to the cloud and was also received back because the application has also

subscribed to the same topic where it publishes the data.

To see the uploaded data from the AWS IoT console navigate to the Test page and subscribe to the topic used by the application:

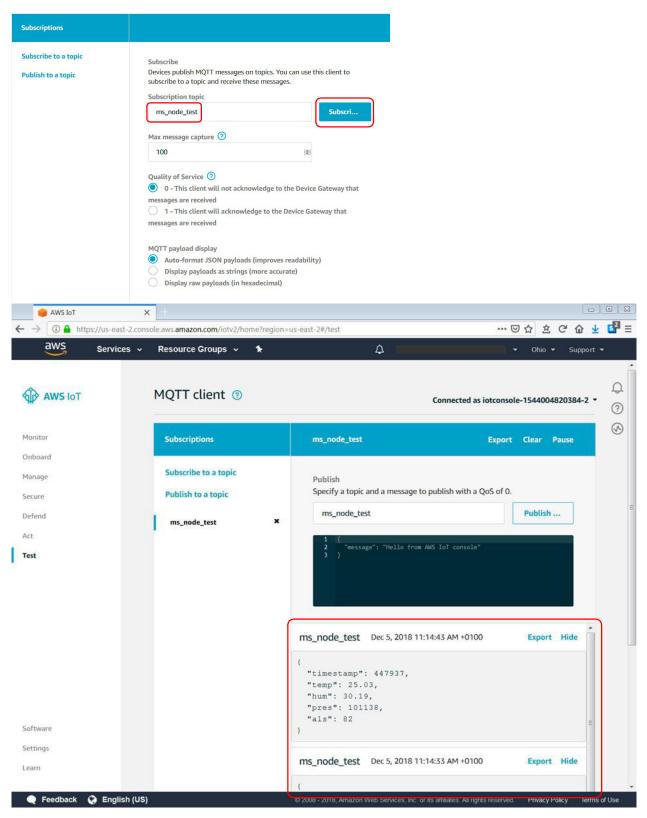


Figure 16.

ON Semiconductor and in are trademarks of Semiconductor Components Industries, LLC dba ON Semiconductor or its subsidiaries in the United States and/or other countries. ON Semiconductor owns the rights to a number of patents, trademarks, copyrights, trade secrets, and other intellectual property. A listing of ON Semiconductor's product/patent coverage may be accessed at www.onsemi.com/site/pdf/Patent-Marking.pdf. ON Semiconductor reserves the right to make changes without further notice to any products herein. ON Semiconductor and separating the suitability of its products for any particular purpose, nor does ON Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation special, consequential or incidental damages. Buyer is responsible for its products and applications using ON Semiconductor products, including compliance with all laws, regulations and safety requirements or standards, regardless of any support or applications information provided by ON Semiconductor. "Typical" parameters which may be provided in ON Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. ON Semiconductor does not convey any license under its patent rights nor the rights of others. ON Semiconductor products are not designed, intended, or authorized for use as a critical component in life support systems or any FDA Class 3 medical devices or medical devices with a same or similar classification in a foreign jurisdiction or any devices intended for implantation in the human body. Should Buyer purchase or use ON Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold ON Semiconductor and its officers, emplo

PUBLICATION ORDERING INFORMATION

LITERATURE FULFILLMENT:

Literature Distribution Center for ON Semiconductor 19521 E. 32nd Pkwy, Aurora, Colorado 80011 USA Phone: 303-675-2175 or 800-344-3860 Toll Free USA/Canada Fax: 303-675-2176 or 800-344-3867 Toll Free USA/Canada Email: orderlit@onsemi.com N. American Technical Support: 800–282–9855 Toll Free USA/Canada

Europe, Middle East and Africa Technical Support: Phone: 421 33 790 2910 ON Semiconductor Website: www.onsemi.com

Order Literature: http://www.onsemi.com/orderlit

For additional information, please contact your local Sales Representative