Technical Article Long Range Sensor-to-Cloud: Connecting to Amazon Web Services with Sub-1 GHz-based Devices



Gaurav Pal

Proven effective through many years of use in proprietary networks, bands within the Sub-1 GHz frequency enable long-range communication for applications such as connected door or window sensors, smoke alarms and metering. In addition to long-range, these radio waves can better penetrate walls and bend around obstacles while consuming very little power for operation.

These attributes make Sub-1 GHz an optimal choice for Internet of Things (IoT) applications that connect sensors to IoT platforms. However, when connecting a system of smart nodes to the cloud, Sub-1 GHz technologies have historically been hindered by a lack of standard networking solutions within this spectrum. As a result, companies typically require staff with low-level RF communication expertise to implement their own connectivity stacks, and, in turn, transfer data from these accessible IP networks to downstream services.

To streamline this process, TI developed the TI 15.4-Stack software, an 802.15.4g communications stack for Sub-1 GHz, along with Linux-based gateway software running on the Sitara[™] AM3358 processor that powers the BeagleBone Black. In addition, TI has taken it a step further by collaborating with stackArmor to develop a cloud-based Software-as-a-Service (SaaS) solution to help engineers connect their applications to the cloud. As an Amazon Web Services (AWS) consulting partner, stackArmor provides cloud migration and support services, and maintains a dedicated IoT cloud solutions practice.

The below example outlines how to move sensor data arriving at a Sub-1 GHz communication stack to the AWS cloud for storage, display and analysis.

AWS Application Overview

AWS is a fully managed cloud service that includes both specific AWS SaaS applications and general cloud computation functionality. Figure 1 works to outline how stackArmor's application uses two SaaS solutions (AWS IoT's Rules Engine and DynamoDB) to transfer data received by the Sub-1 GHz gateway to the cloud. For this example, the referenced device is a BeagleBone Black-based gateway.

To visualize the data for this application, a popular opensource platform, Angular.js, was used to create a dashboard powered by EC2 – another AWS service that enables server instances to run non-AWS software. EC2 was also used to run the NGINX security proxy for malware protection along with other ancillary services.



Figure 1. StackArmor AWS Application

1



As shown in Figure 1 above, the AWS IoT service first establishes communication between a device and the cloud. However, for the AWS IoT server to receive messages, two things must be installed – a software development kit (SDK) and a security certificate (both are provided by AWS).

The AWS SDK is available in various languages, including C and Node.js (JavaScript), but assumes the underlying hardware supports a TCP/IP stack. For this reason, it cannot be run directly on non-IP nodes such as those running BLE or Sub-1 GHz. We chose the Node.js SDK for our development.

Additionally, the SDK provides necessary commands for connecting and sending as well as receiving MQTT messages from the AWS IoT cloud service. This functionality is perfect for our application as it requires bi-directional transmission of sensor data from the device to the cloud, along with actuation signals that change the state of the device or sensors through the web application.

The associated security certificate must also be installed to ensure that only authorized devices can connect to the cloud.

To function properly, the AWS IoT service also requires setup and configuration of a cloud representation of our BeagleBone Black-based gateway, which allows messages sent from the physical gateway through the AWS service to be published on this digital representation of the gateway. And as AWS IoT uses MQTT, we created corresponding topics to represent the actual end-node sensors and actuators located on the SimpleLink[™] CC1350 LaunchPad[™].



SimpleLink™ CC1350LaunchPad™

BeagleBone Black development board



Connected to the "digital gateway" are AWS IoT's Rules Engine and a DynamoDB (NoSQL) database that stores messages received from the physical device. Rules Engine is used to determine where these messages are stored, while DynamoDB actually stores the messages in either a network-specific or end-node file.

To make the file determination, Rules Engine reviews the topic associated with each message and then sorts it to the appropriate file. Unlike embedded applications that require specific actions to be coded, many AWS actions can simply be specified by configuration. For example, you can configure AWS IoT to send an SMS message or email when a specific message arrives. Because we chose Rules Engine and DynamoDB for this application, the set-up did not involve any coding.

As previously mentioned, the IoT dashboard was developed with Angular.js, which periodically sends queries to the DynamoDB database for new data. This data is then displayed on the dashboard using easy-to-read charts and graphs. This web-based application can also transmit actuation signals from the dashboard to the gateway, through the AWS cloud connection, to toggle network power or manipulate the gateway sensors.

To learn more about this implementation, including in-depth documentation of the hardware set-up and communications stack, you can check out the TI reference design or download the Sensor to cloud reference design to get started.

2

TEXAS INSTRUMENTS www.ti.com

You May Also Order Kits Here:

- SimpleLink CC1350 LaunchPad™ development kit
- BeagleBone Black

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES "AS IS" AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI's products are provided subject to TI's Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI's provision of these resources does not expand or otherwise alter TI's applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265 Copyright © 2023, Texas Instruments Incorporated