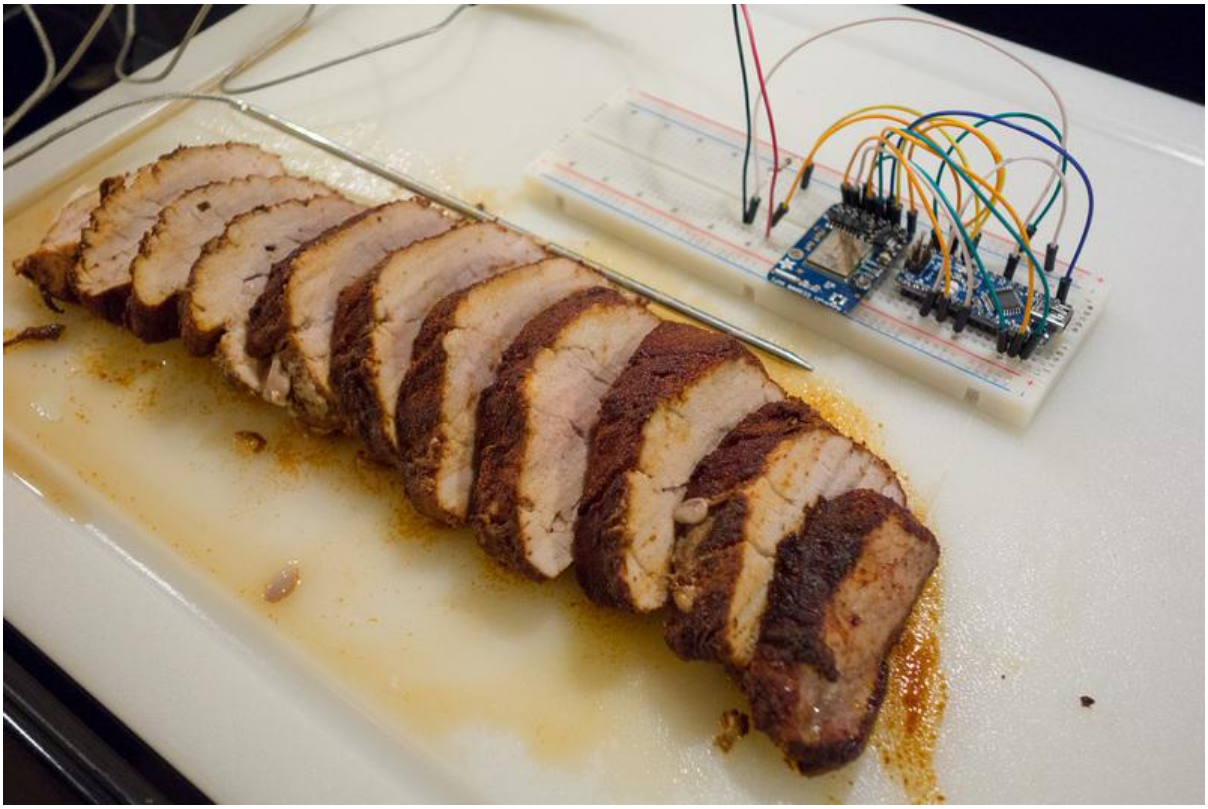
Cloud Thermometer

Created by Tony DiCola

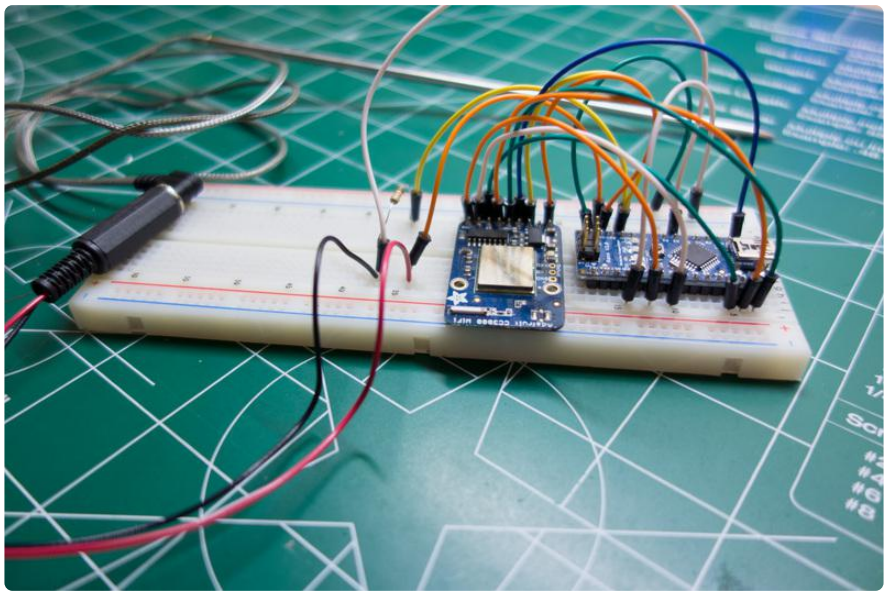


https://learn.adafruit.com/cloud-thermometer

Last updated on 2023-08-29 02:26:17 PM EDT

Table of Contents

Overview



Have you ever wished you could know when something you put in the oven was done cooking? Maybe not 10 minute cookies but how about the 3 or 4 hour turkey roast? You might have used a gadget like a kitchen probe thermometer () to monitor the temperature of food in the oven. Probe thermometers are great at measuring the temperature of cooking food, however there are a couple ways they can be improved. First, most probe thermometers (the inexpensive ones at least) can't be monitored remotely, so you need to be in the kitchen constantly watching them. Second, these thermometers only display the current temperature and don't give any prediction of when the food will be at a desired temperature.

This guide will show you how to build a thermometer that solves both the problems above. By using an Arduino () and Adafruit CC3000 WiFi breakout (http://adafru.it/1469), you can build a probe thermometer that logs temperature data to Amazon's DynamoDB () cloud database service. With the temperature data in the cloud, you can monitor it remotely over the web, and even use the history of measurements to predict when the food will be ready!

Before you start it will help to familiarize yourself with the following guides:

- CC3000 Setup ()
- Thermistor ()

You will also want to download and extract the two Arduino sketches in the following download link. These sketches will be used later in the guide to calibrate and run the thermometer hardware.

Download Cloud Thermometer Sketches

Why DynamoDB?

The motivation to use Amazon's DynamoDB for the cloud thermometer is driven by a few reasons:

- Need a data store that supports appending new measurements to existing data. Because the Arduino only has a few kilobytes of memory, it's not feasible for the Arduino to download and update the entire history of temperature data. Unfortunately this rules out most file or blob storage services like Amazon S3 (), Google cloud storage (), or Azure storage () because they don't support appending to existing documents.
- Need a data store that can be accessed without HTTPS. With only about 30 kilobytes of program space available, it's not possible to support a cloud service that requires secure HTTPS communication. Unfortunately this rules out Dropbox's blob and structured storage services ().
- Prefer a data store with simple APIs and authentication. DynamoDB has a somewhat complex signing process that requires multiple SHA256 hash computations. However this signing process is simpler than other data stores which require calling into separate authentication services or more to authenticate.
- Prefer a data store that can be accessed directly from a web browser. This isn't a strict requirement, but it simplfies the visualization of temperature data by not requiring the use of an intermediary web service. Luckily DynamoDB support is a part of the recently released AWS browser javascript SDK ().

Continue on to learn about the hardware needed to build the cloud thermometer.

Hardware

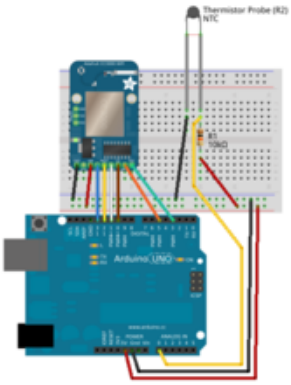
Parts

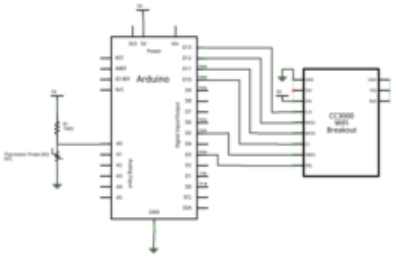
For this project you will need the following hardware:

- Arduino Uno (http://adafru.it/50), Mega (http://adafru.it/191), or Nano.
- CC3000 shield (http://adafru.it/1491) or breakout (http://adafru.it/1469).
- Thermistor temperature probe, like what can be found in a simple kitchen probe thermometer. For example this probe () should work for the project. You could

also build a probe out of a thermistor (http://adafru.it/372) and food-safe metal tube, but you will need to very careful that all parts of the probe are food and oven temperature safe. It will be easier and cheaper to use an off the shelf kitchen thermometer probe.

- A 2.5mm mono jack, or alligator clips (http://adafru.it/321) for attaching the temperature probe to your hardware.
- One 10 kilo-ohm 1/4 watt resistor.
- Hook-up wire (http://adafru.it/1311) and a breadboard (http://adafru.it/64) or perf board (http://adafru.it/571).
- Power source such as a 9 volt battery (http://adafru.it/67), 6x AA batteries (http://adafru.it/248), or a wall wart (http://adafru.it/63). Note: A 9 volt battery will only power the thermometer at most for a couple hours continuously.



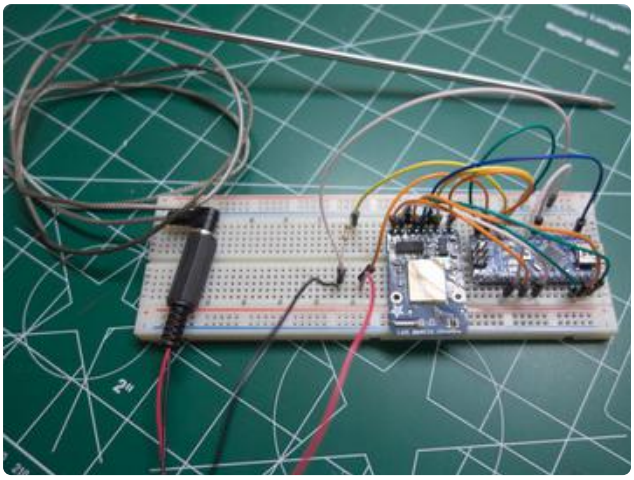


Assembly

For the CC3000, connect it to the Arduino in the same way as this CC3000 tutorial ():
Arduino 5V to CC3000 VIN
Arduino ground to CC3000 ground
Arduino digital pin 13 to CC3000 CLK
Arduino digital pin 12 to CC3000 MISO
Arduino digital pin 11 to CC3000 MOSI
Arduino digital pin 10 to CC3000 CS
Arduino digital pin 5 to CC3000 VBEN
Arduino digital pin 3 to CC3000 IRQ

Next, connect one end of the 10K resistor to 5V power on the Arduino, and the other end to one lead from the thermistor probe. Connect the remaining thermistor lead to ground on the Arduino. Finally connect a wire in between the resistor and thermistor lead connection, and run the wire to an analog input on your Arduino (like analog input 0).

If you don't have a 2.5mm jack for your probe thermometer, you can connect alligator clips to each metal terminal on the plug. See the photo on the left for an example alligator clip connection. You can also see an example of the hardware I built using an Arduino Nano and CC3000 breakout.



Continue on to learn about calibrating the thermistor probe.

Calibration

Before you can run the cloud thermometer sketch you will first need to calibrate your thermistor. This calibration process will produce three coefficient values that are necessary for determining the temperature of the thermistor probe.

To calibrate the thermistor you need the following:

- The assembled hardware from the previous step.
- A glass of ice water, luke-warm water, and hot water.
- A thermometer you can use to measure the temperature of the water.

Note: Make sure the hardware is fully assembled with the CC3000 and Arduino before calibrating. Adding parts to the hardware later might make the calibration inaccurate.

Load the ThermistorCalibrate sketch in Arduino and upload it to your hardware. Open the serial port (at 115200 baud) and follow the instructions from the hardware to calibrate the thermistor. You will place the thermistor into the different temperature water glasses and tell the hardware the temperature of the water. When calibration is complete you will want to save the 3 thermistor coefficient values (A, B, and C) so they can be used in the cloud thermometer sketch.

You can watch the video below to see an example of calibrating the thermistor:

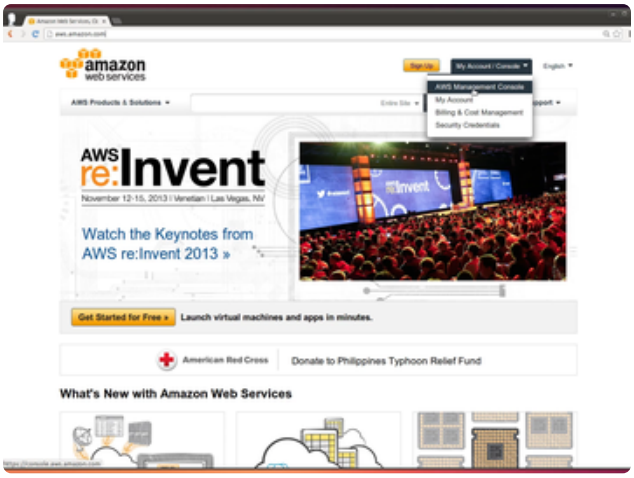
Continue on to setup the DynamoDB table used to store temperature data.

DynamoDB Table Setup

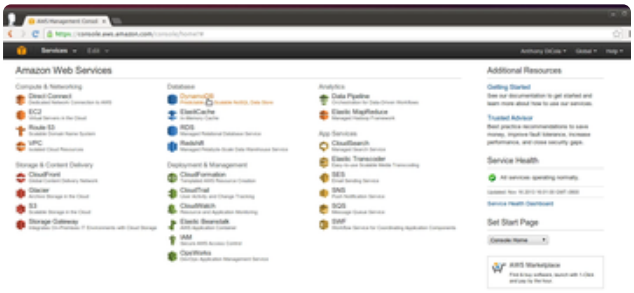
In this step you'll set up the table that will store temperature data in Amazon's DynamoDB service. Before you begin you'll want to create an Amazon web service account if you don't have one already. To create an account you can follow the steps under 'Sign up for the Service' at this link ().

Once you have an AWS account, the video below will show you how to setup the table:

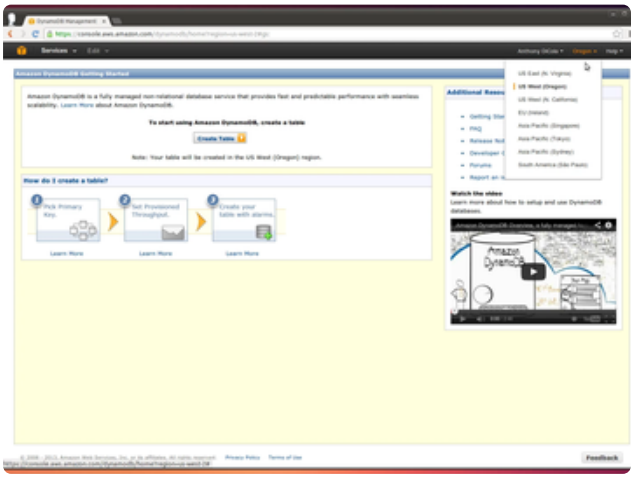
Here's a summary of the steps shown in the video:



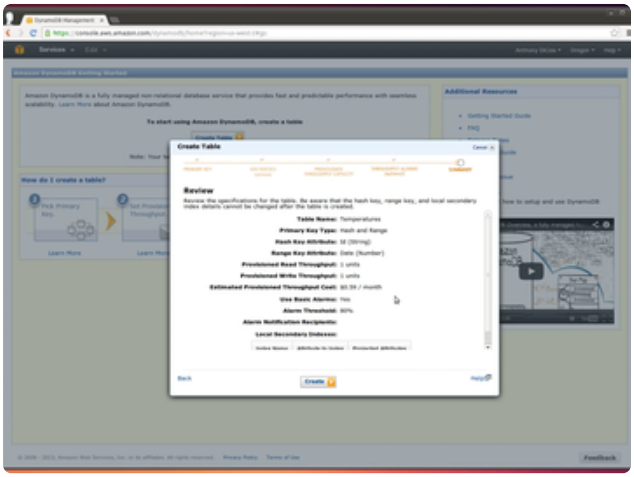
After creating an AWS account (and noting what your access key and secret access key values are as they will be used later in the configuration), navigate to http://aws.amazon.com/ () and select the AWS management console link in the upper right corner.



Select the DynamoDB service.

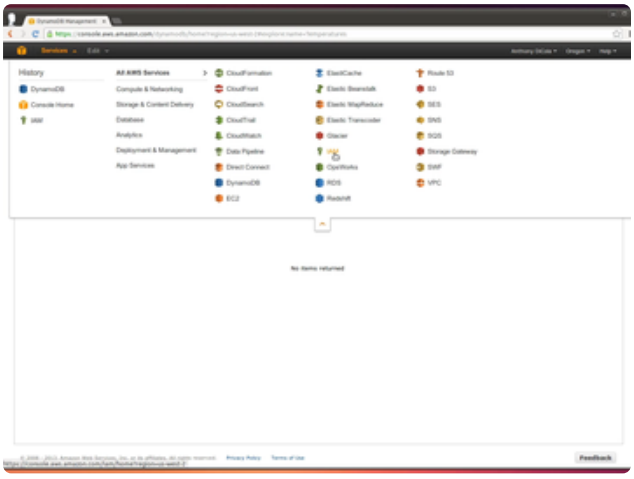


Note in the upper right corner you can select the region where the table will be created. Pick a region that is near your location for the best performance, then click Create Table to open the table creation wizard.



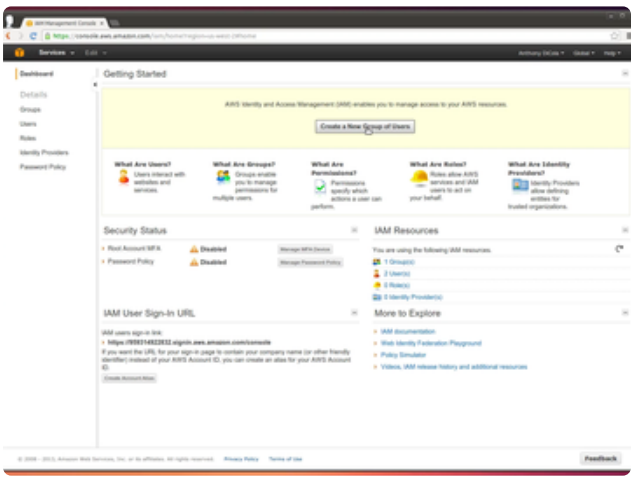
Follow the wizard to create a table with the following attributes:
Name: Temperatures
Primary Key Type: Hash and Range
Hash Attribute Type: String
Hash Attribute Name: Id
Range Attribute Type: Number
Range Attribute Name: Date
No secondary indexes.
Default values for provisioned throughput capacity and throughput alarms
After finishing the table creation wizard, refresh the tables list until the new table has a status of 'ACTIVE'.

Also take note of the Amazon Resource Name (ARN) in the 'Details' tab for the table. This ARN value will be used later in the table setup.

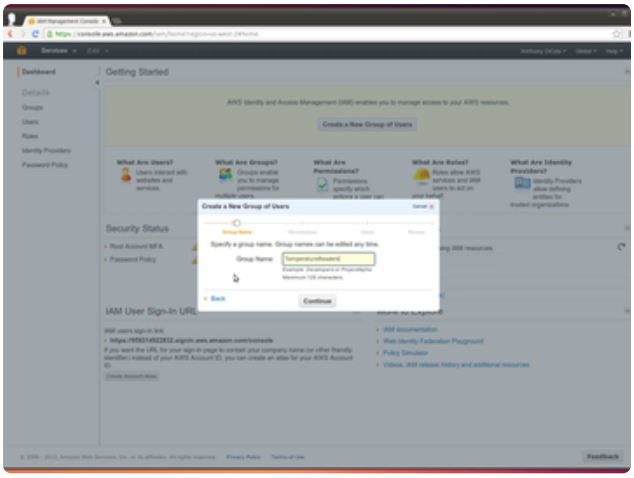


Next you will setup a group and user with read only access to the table. This will allow you to access the table from a website without embedding your personal AWS access codes in the source (where they are visible to anyone who views the web page).

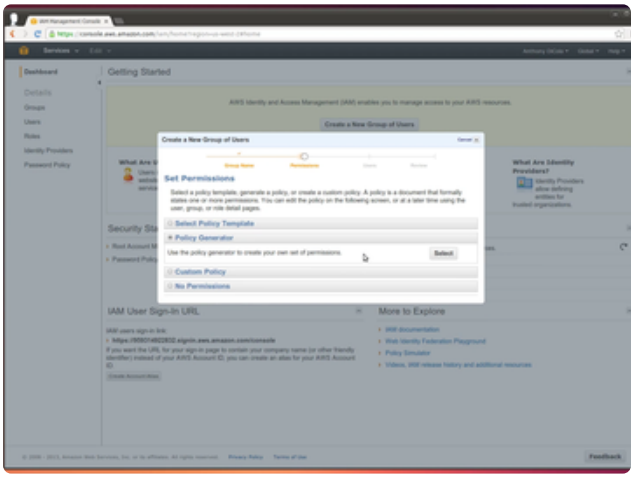
In the Services menu at the upper left, select the 'IAM' service.



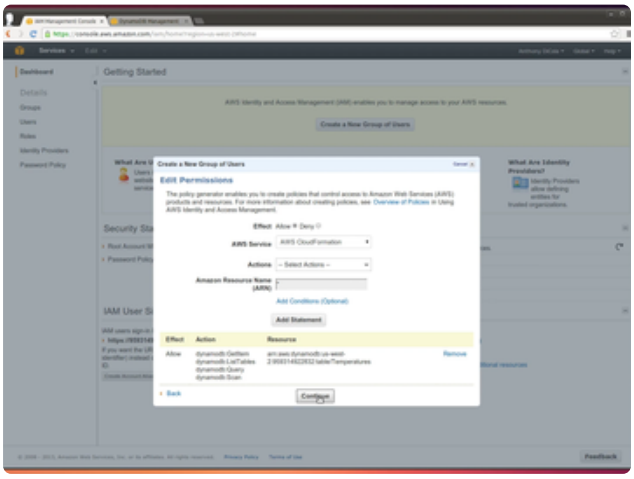
Click 'Create a New Group of Users' to open the group and user creation wizard.



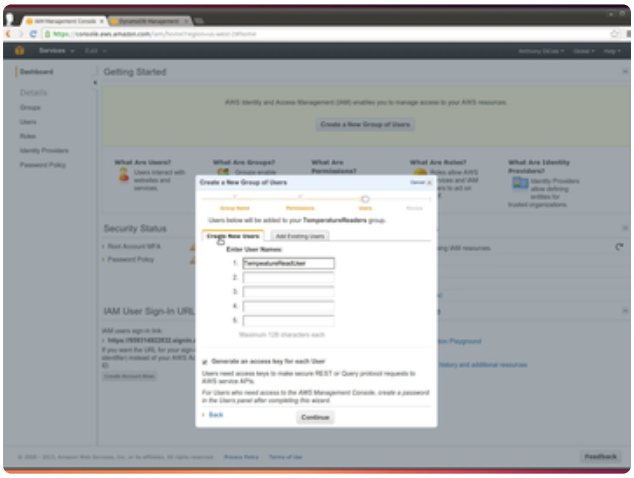
Set the group name to 'TemperatureReaders'.



Select the 'Policy Generator' option to set permissions.



Add a policy statement with the following attributes:
Effect: Allow
AWS Service: DynamoDB
Actions: GetItem, ListTables, Query, Scan
Amazon Resource Name (ARN): The ARN value for the table you created earlier.
Click continue to see a summary of the generated policy, then continue again to create a user.



In the 'Create New Users' tab, create a new user with name 'TemperatureReadUser'. Make sure the 'Generate an access key for each User' is selected, then click continue.

After reviewing the details, click continue again. When the user is created, open the 'Show User Security Credentials' drop down and take note of both the access key and secret access key values for this read only user.

Continue on to learn how to set up the sketch for the cloud thermometer.

Software Setup

Finally in this step you'll configure the cloud thermometer sketch, run it, and learn how to visualize the temperature data with a web page. Before you begin make sure you have completed all the previous steps and have the following ready:

- CloudThermometer Arduino sketch, downloaded from the overview page ().
- Assembled cloud thermometer hardware.
- Three Steinhart-Hart coefficient values from the calibration ().
- Access key and secret access key of your AWS account.
- Access key and secret access key of the read only user created in the table setup ().
- Computer with Python () 2 or 3 installed. (note for Python 3, the actual command to run an HTTP server is different from what's mentioned in the video ()).

You can watch the video below to see the setup and usage of the cloud thermometer:

For a summary of the setup, you will need to load the CloudThermometer sketch in Arduino and change the define values at the top of the sketch as follows:

- ADAFRUIT_CC3000_IRQ, ADAFRUIT_CC3000_VBAT, ADAFRUIT_CC3000_CS: Set these to the appropriate pins which are connected to the IRQ, VBEN, and CS pins respectively on the CC3000.
- WLAN_SSID, WLAN_PASS, WLAN_SECURITY: Set these to the appropriate values for your wireless network.
- THERMISTOR_PIN: The ADC pin which is connected to the thermistor.

- SERIES_RESISTOR: The value (in Ohms) of the resistor connected in series with the thermistor. This should be 10,000.
- ADC_SAMPLES: Keep the default value of 5.
- A_COEFFICIENT, B_COEFFICIENT, C_COEFFICIENT: Set these to the three values that were generated in the calibration step.
- TABLE_NAME: Set this to the name of the DynamoDB table, Temperatures.
- ID_VALUE: Set this to a unique string value that identifies this set of temperature measurements. For example the name of the food you're cooking is a good value to use for the ID value. Change this value each time you start a new temperature measurement. The ID value will be used to identify and select a temperature measurement in the web page.
- AWS_ACCESS_KEY, AWS_SECRET_ACCESS_KEY: Set these to the values of your personal AWS account. Do not use the credentials for the read only user here as these are the credentials that are used to write data to the table!
- AWS_REGION: Set this to the region value from table in the link in the comment. You will want to select the region where your DynamoDB table was created.
- AWS_HOST: Set this to the endpoint URL from the table in the link in the comment.

Keep the defaults for the remaining configuration values. Load the sketch on your hardware and watch the serial monitor (at 115200 baud) for output from the hardware.

After connecting to your wireless network you should see output every minute as the sketch sends data to DynamoDB. In particular look for an 'HTTP/1.1 200 OK' response value to ensure the hardware is successfully writing to DynamoDB.

If you see an error like 'HTTP/1.1 403 Forbidden' or HTTP/1.1 400 Bad Request', double check your access keys, region, and host configuration is correct and try again.

Once the hardware is running you can setup a simple web page to visualize the data. Edit the index.html file in the webpage subdirectory of the CloudThermometer sketch folder. Add the read only user's access key and secret access key values in the appropriate spot noted in the comments. Also set the region value to the same region value as used in the cloud thermometer sketch.

Save the updated index.html and navigate to the folder in a terminal window. If you have Python 2 installed, enter the command:

python -m SimpleHTTPServer

Otherwise if you have Python 3 installed, enter:

python3 -m http.server

This will run a small HTTP server to serve the index.html page to your web browser. Open http://localhost:8000/ () in a browser to see the cloud thermometer web page. Select your measurement ID value to view the temperature graph.

Continue on to see ideas for future enhancements to the project.

Future Work

This project is a great example of data logging directly to a cloud web service. You can extend this project in a lot of interesting ways, such as:

- Add more thermistor probes or different types of sensors to log more data. To do this look at modifying the payload sent to DynamoDB so it has more columns of sensor data.
- Create multiple instances of the hardware, all logging to the same table. DynamoDB is quite scalable and should be able to support hundreds or even thousands of sensors writing data.
- Use the signature generation code in this sketch to call other Amazon web services. For example Amazon has services for storing file data (S3 ()), asyncronous communication (SQS ()), notifications (SNS ()), and more.

Have fun with the project!