CleanSlate

IOT AS **EASY AS PI**

BY DARREN MILLS

CLARITY IN TECHNOLOGY

ABOUT **CLEANSLATE**

WHO **WE ARE**

Founded in 2000, we are a privately held company, headquartered in Carmel, Indiana. Voted "Best Places to Work in Indiana" from 2016 through 2018

CleanSlate dedicated practices:
Cloud, Enterprise Integration, Salesforce

Our team has a combined total of 49 years of cloud experience. Our heads are in the clouds...

Join us on Slack:
#sponsor-cleanslate

OUR **SPEAKER**

DARREN **MILLS**
Director of Technology

AWS Certified • Salesforce Certified
MuleSoft Certified • Java Certified
New Relic Certified

Fun Fact – Licensed pilot and I like to watch scary movies

OUR **PARTNERS**

salesforce · MuleSoft · aws · docker · Microsoft Azure

SESSION **AGENDA**

01

Internet of Things Overview

General overview of IoT to understand how it fits in today's market and the common challenges of a successful implementation.

02

AWS IoT Services Overview

Learn about the AWS IoT Services that can enable your business to quickly and securely implement IoT enabled products.

03

AWS IoT Raspberry Pi Weather Station

Learn about the components of a Raspberry Pi that comprise of the Weather Station and how they fit within the AWS IoT architecture to collect its statistics.

04

AWS IoT Weather Station Demo

Let's walk through the configuration steps required to connect an IoT device to the AWS IoT platform and configure secure communication between the gateway device and your AWS endpoint.

INTERNET OF THINGS (IOT) **OVERVIEW**

- Internet of Things (IoT) is a system of devices connected to the Internet with the ability to collect and exchange data from users or environment with no human intervention.

- The device or “thing” in IoT could be any type of device embedded with specialized software to interact with its electronics or sensors.
  - Home Automation / Amazon Alexa / Google Nest / Smart Lights
  - Security System / Doorbells
  - Health Monitors
  - Automobile Diagnostics
  - Government / Traffic Lights

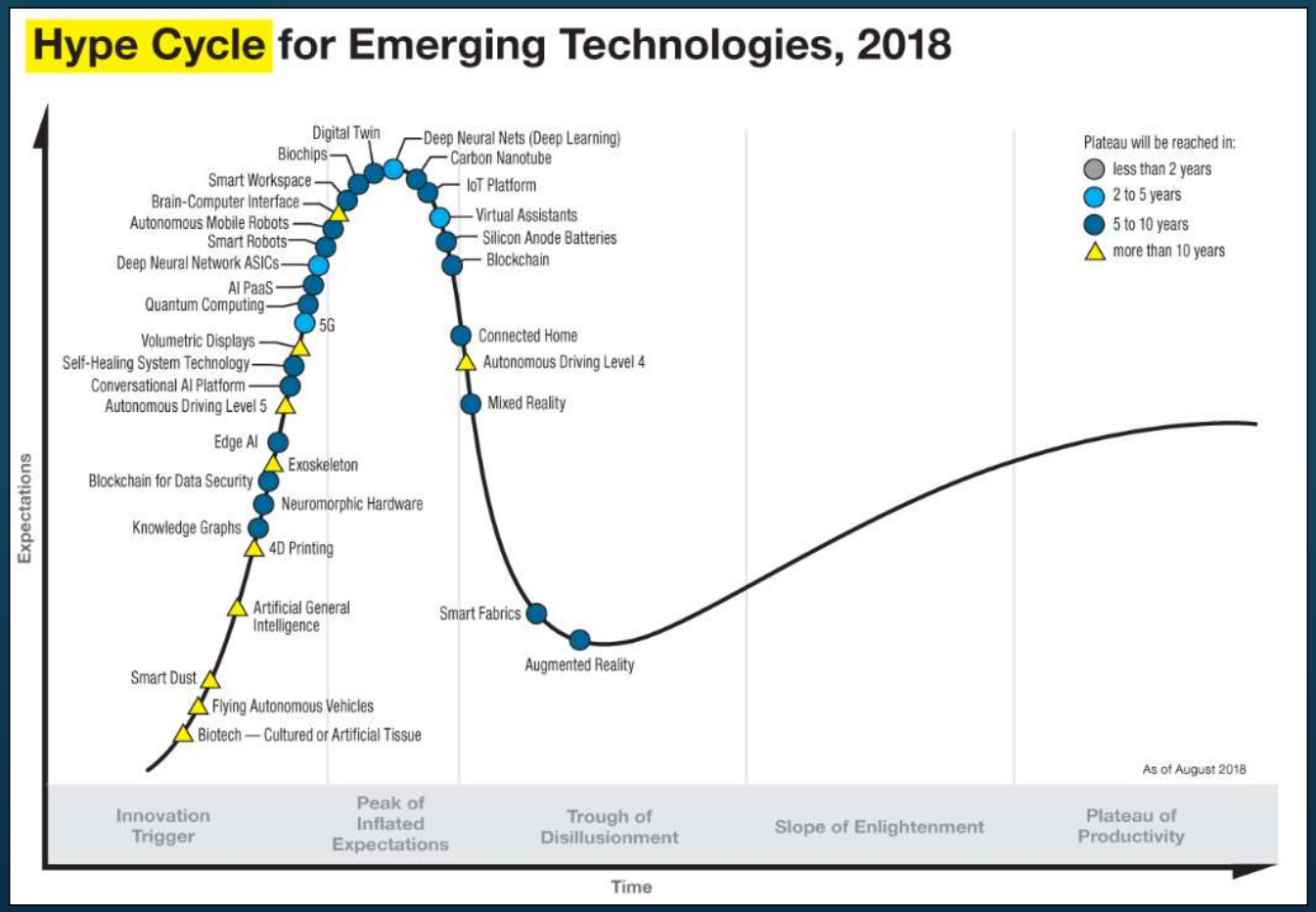
IOT **FUN FACTS**

- The first IoT device was a Coke vending machine at Carnegie Mellon University in 1982. It was connected to the internet and reported the inventory and temperature of the drinks. It was later followed up in 1989 with an Internet managed Toaster for a conference, by John Romkey.

- The term “Internet of Things” was coined in 1999 by Kevin Ashton from MIT, by accident when used as the title of a presentation. He later thought “Internet for Things” was a better fit, but the name had stuck.

- Currently, there are more IoT devices connected to the Internet than the World population

IOT **IN THE MARKET**



"Things" Deployed

- 2018 – 7 Billion
- 2019 – 26 Billion
- 2020 – 31 Billion

Top Industries (2020)

- Utilities
- Security
- Healthcare
- Automotive
- Consumer (Home)

IOT **CHALLENGES**

Implementation is difficult due to the complex nature of the different components in the ecosystem of IoT

Communication	Security	Presence Detection	Power Consumption	Bandwidth
• Devices may be talking to a server to collect data, or the server may be talking to the devices, or maybe those devices are talking to one another. • Data needs to get from point A to point B quickly and reliably.	• **Authorization**: It's essential to make sure that the IoT device and endpoint has proper authorization to send or receive the stream of data. • **Open Ports**: Devices are vulnerable when listening to an open port out on the Internet. • **Encryption**: Data should be encrypted between devices and endpoints.	• It's important to know when an IoT device goes offline and when that device comes back online.	• IoT devices signaling and sending data between one another takes a toll on power and CPU consumption. • With all this communication, you need minimal battery drain and low power consumption.	• Bandwidth on a cellular network is expensive. Especially with hundreds of thousands of IoT devices on a network sending request/response signals to your endpoint.

IOT CHALLENGES - **DATA**

"One of the myths about the Internet of Things is that companies have all the data they need, but the real challenge is making sense of it."
– Chris Murphy, Information Week

Volume	Security	Privacy	Analytics
• Large amounts of data continually being gathered in real-time from thousands/millions of devices simultaneously • IoT data expected to double every two years, which requires large flexible storage solutions	• Ensure data is secure both in transit and at rest through encryption and key management • Ensure the integrity of the data from devices to ensure the device has not been compromised	• Removal of PII data to protect the identities of the consumers • Lack of data classification standards to protect confidential data • Legal ramifications of large data collection and data retention • Public perception of eavesdropping and profiling	• Data Analytics is the key to making enhanced business decisions and efficiency gains • Difficult to consolidate data which is a mixture of structured and unstructured information across the enterprise • Too much data to analyze thus requires some form of AI/ML to provide insight into future business decisions

AMAZON **IOT SERVICES**

DEVICE SOFTWARE

- FreeRTOS
- AWS IoT Greengrass

CONTROL SERVICES

- **AWS IoT 1-Click Service**
- **AWS IoT Core**
- **AWS Device Management**
- **AWS Device Defender**

ANALYTICS SERVICES

- AWS IoT Analytics
- AWS IoT Events
- AWS IoT SiteWise
- AWS IoT Things Graph

AWS IOT **1-CLICK SERVICE**

AWS IoT 1-Click is a service that enables simple devices to trigger AWS Lambda functions that can execute an action

- Easily trigger Lambda functions from supported devices
- Secure connectivity right out of the box
- Easy to manage supported devices

Enterprise Dash Button

- Implemented via the AWS IoT 1-Click Service
- 3 Different Button Actions
  - Single Click
  - Double Click
  - Long Press

AWS **DASH BUTTON HISTORY**

- Introduced on March 31, 2015 – originally believed to be an April Fools joke
- Over 250+ Dash Buttons were introduced
- Amazon's first IoT Device targeted to quickly replenish household items
- Helped Amazon learn buyer characteristics to augment their AmazonFresh grocery service
- Discontinued on August 31, 2019
  - Amazon Alexa replaced functionality
  - Consumer Rights violations because price was not known by end user upon ordering

AWS **IOT DEVICE SOFTWARE**

AWS IoT Device Software provides the essential building blocks to create cloud managed IoT devices.

- FreeRTOS
  - Open source, real-time operating system for microcontrollers that makes small, low-power edge devices easy to program, deploy, secure, connect, and manage
- IoT Greengrass
  - Extends AWS to edge devices so they can act locally on the data they generate, while still using the cloud for management, analytics, and durable storage.
- Amazon Common Software (ACS) – *released in Feb. 2020, still in preview*
  - Optimized software for integrating Amazon Device SDKs on your devices
    - Frustration-Free-Setup (FFS) – making setup easier for the customer
    - Pre-validated and memory optimized components for common functions such as connectivity

AWS **IOT CORE SERVICE**

AWS IoT Core is a managed cloud service that lets connected devices easily and securely interact with cloud applications and other devices.

- Connect and Manage your Devices
  - Supports HTTPS, WebSockets and MQTT and custom protocols
- Secure Device Connections and Data
  - Automated configuration and authentication of device
  - Encryption provided End-to-End
- Process and Act upon Device Data
  - Allows simple ways to filter, transform and act upon device data
  - Support business rules that interact with AWS Services like Lambda, Kinesis, S3, DynamoDB, etc.
- Read and Set Device State
  - Allows the device to appear online to your application, even if the device is temporarily offline
- Scale to Billions of Devices
  - Framework is built to support billions of devices and trillions of messages

AWS **IOT DEVICE MANAGEMENT**

AWS IoT Device Management makes it easy to securely register, organize, monitor, and remotely manage IoT devices at scale.

- Fast Device Registration
  - IoT Registry provides ability to securely manage a large fleet of devices quickly
- Simple IoT Device Organization
  - Allows your devices to be organized into groups and manage policies at a group level
- Locate Connected Devices Quickly
  - Provides search capabilities to find devices in your fleet based upon ID, state, attribute, etc. quickly
- Easy Remote Device Management
  - Allows you remotely update the software running on your devices after they have been deployed in the field, ensuring your devices are always running on the latest software version

AWS **IOT DEVICE DEFENDER**

AWS IoT Device Defender is a fully managed service
that helps you secure your fleet of IoT devices.

- Audit Device Configurations
  - Audits IoT configurations associated with your devices against a set of defined IoT security best practices
- Monitor Device Behavior
  - Detects anomalies in device behavior that may indicate a compromised device
  - Allows you to define security rules:
    - Open Ports
    - Communication Routes (*who can the device talk to and who can talk to the device*)
    - Data Volume
- Publish Security Alerts
  - Publishes security alerts to the when an audit fails or when behavior anomalies are detected
    - AWS IoT Console
    - Amazon CloudWatch
    - Amazon SNS

AWS **IOT ANALYTIC SERVICES**

AWS IoT Analytics is a fully-managed service that makes it easy to operationalize analytics on massive volumes of IoT data.

- Prepares your IoT Data for Analysis
  - Integrated with IoT Core to easily ingest device data
  - Provides capabilities to filter and transform data
  - Ability to integrate data from external sources with Lambda functions
- Easily run Queries on IoT Data
  - Allows simple and ad hoc SQL queries to be executed on your data store
- Data Storage Optimized for IoT
  - Data Store is optimized for time-series to deliver fast response times on IoT queries
- Tools for Machine Learning
  - Apply machine learning to your IoT data with hosted Jupyter notebooks

IOT CHALLENGES - **SOLVED**

Problem	Solution	Solved?
Communication	• AWS IoT Core (HTTPS, WebSockets, MQTT)	✓
Security	• AWS IoT Core (X.509 client certificate) • AWS IoT Device Defender • Amazon Cognito	✓
Presence Detection	• AWS IoT Greengrass • AWS IoT Core (Shadow Devices)	✓
Power Consumption	• AWS FreeRTOS	✓
Bandwidth	• AWS IoT Core (HTTPS, WebSockets, MQTT)	✓
Data Volume	• AWS S3 • AWS RDS • AWS DynamoDB	✓
Data Security	• AWS IoT Core (X.509 client certificate) • AWS IoT Device Defender	✓
Data Privacy	• AWS IoT Device Defender • Amazon Macie (PII Data Detection)	✓
Data Analytics	• AWS IoT Analytic Services	✓

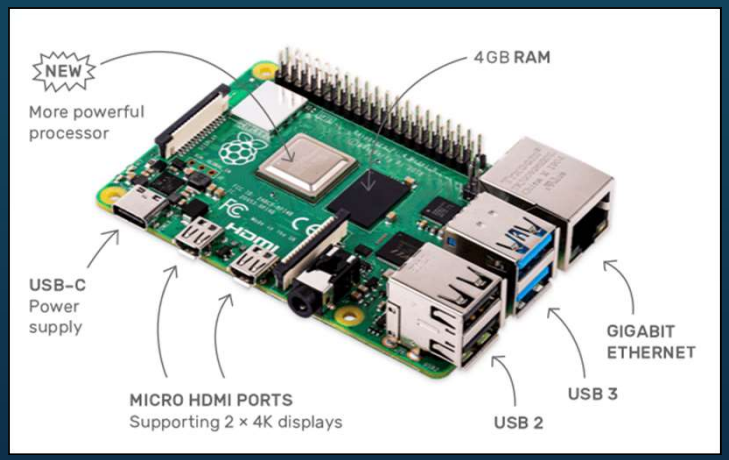
AWS **IOT WEATHER STATION**

RASPBERRY PI **WEATHER STATION**

Raspberry Pi 4 Model B

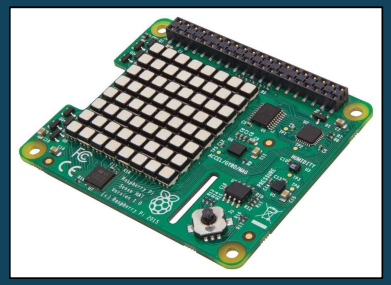
Serves as the IoT Gateway device which is registered with the AWS IoT Platform.

- 1.5 GHz 64-bit quad-core ARM processor
- 4GB RAM
- USB 2 / USB 3 Support
- Two 4k Micro HDMI Display Ports
- GB Ethernet
- USB-C Power Supply



Raspberry Pi Sense HAT

The Sense HAT is an add-on board to provide different weather instrumentation. Launched to ISS in 12/2015.



Sensors:

- Temperature
- Humidity
- Barometric Pressure
- Gyroscope
- Accelerometer
- Magnetometer

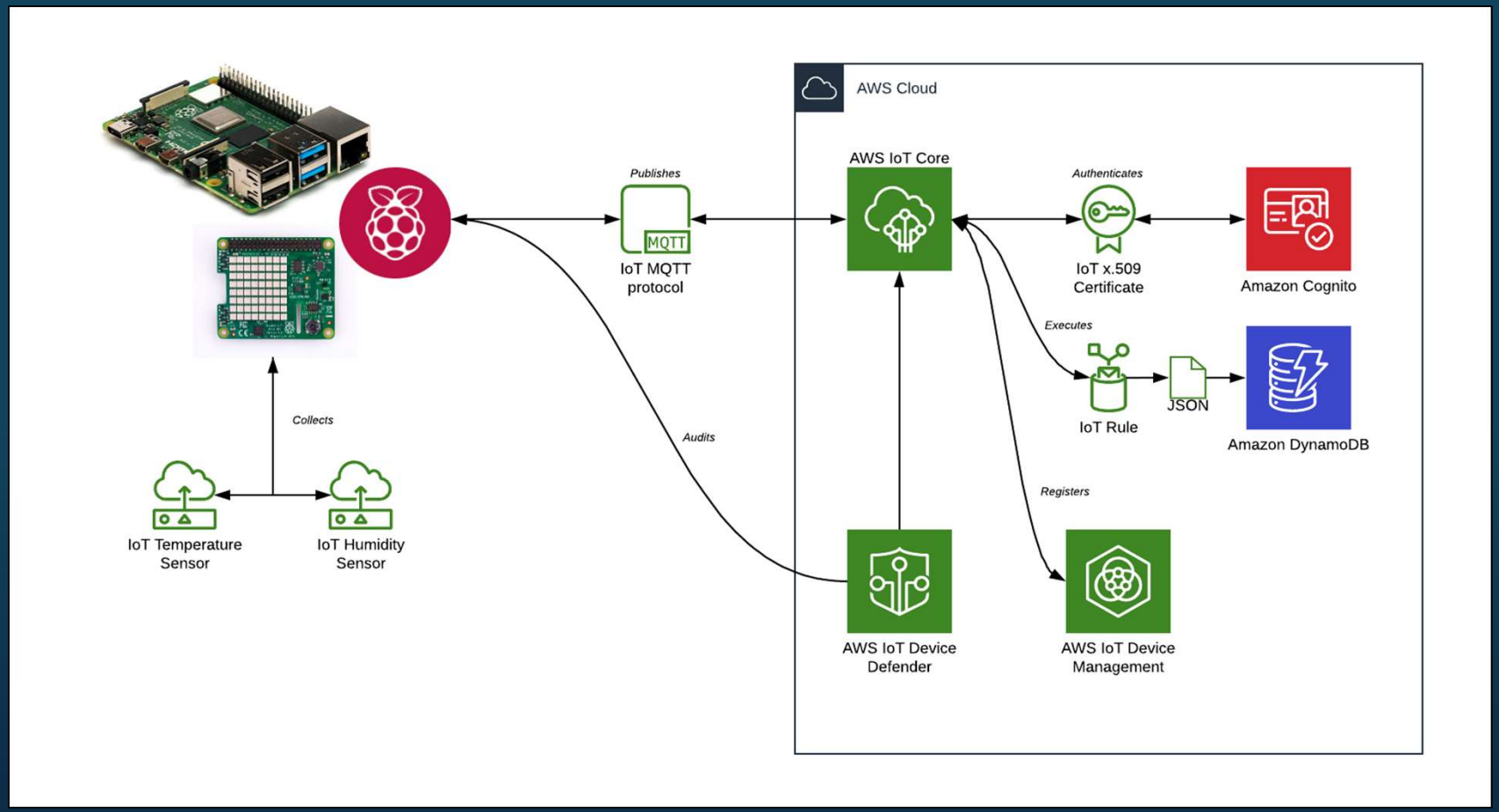
Hologram Nova Cellular Modem

A cellular modem purpose-built for IoT development.

Models Support:

- 2G, 3G and LTE cellular coverage
- External Antenna
- Python SDK

WEATHER STATION **ARCHITECTURE**



IOT **GETTING STARTED**

01

Register Your Thing

Register your device with the AWS IoT Core Framework to allow it to be managed by the platform.

02

Secure Your Thing

Secure your device by generating the required certificates and policies to allow secure communication between your gateway and the platform. In addition, we will register our device with AWS Cognito for Unauthenticated Identities.

03

Develop and Deploy Your Thing

Develop the require code to support your functionality with the AWS SDK for IoT.

04

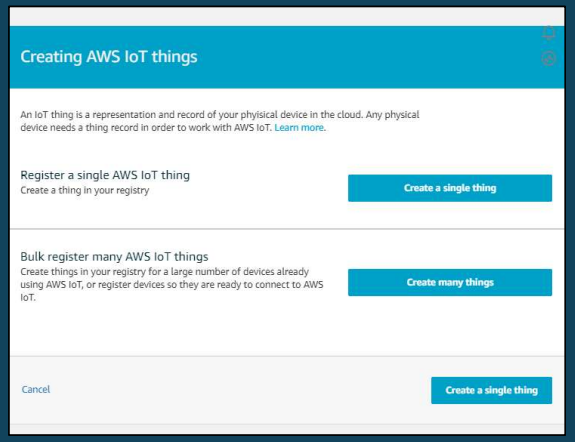
Test Your Thing

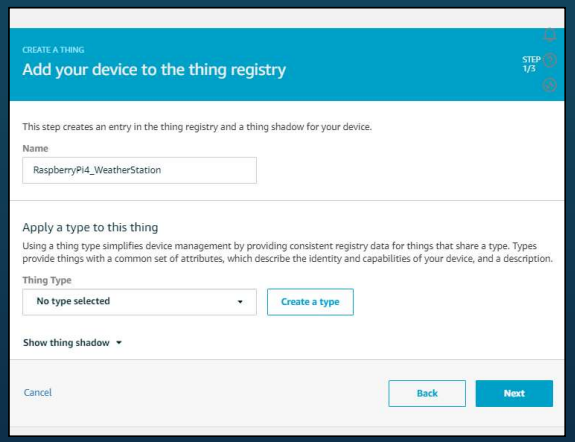
Utilize the built-in testing tools provided with the AWS IoT platform to ensure two-way communication with your device.

REGISTER YOUR **THING**

Pre-Steps

- Logon to the AWS Console





1. Search for the IoT Core Service
   Select “Manage” and then “Register a Thing”
   Click the “Create a single thing” button

2. Name your Thing
   Name: RaspberryPi4_WeatherStation

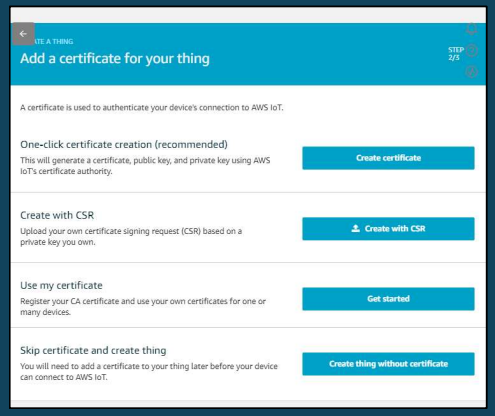
REGISTER YOUR **THING**

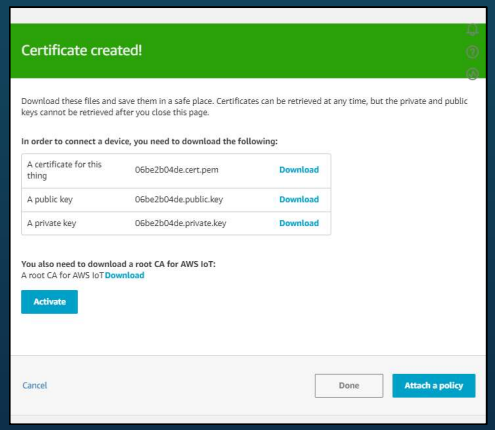
3. Click the “One-click certification creation” button

4. The AWS IoT Platform will create 3 new certificates for your Thing, plus provide you the root CA for AWS IoT.
   - Public Key
   - Private Key
   - Thing Certificate Key
   - Root CA for AWS IoT

   These keys should all be downloaded for future use when setting up your Thing

5. Click the “Activate” button

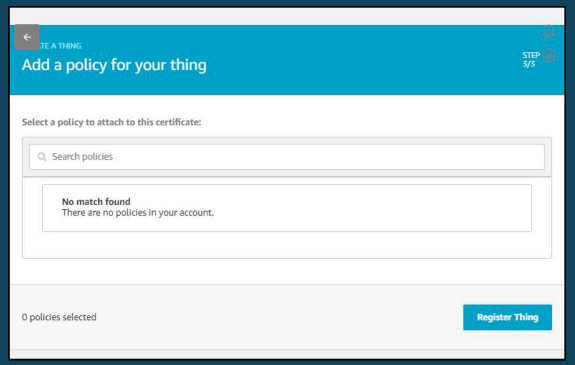


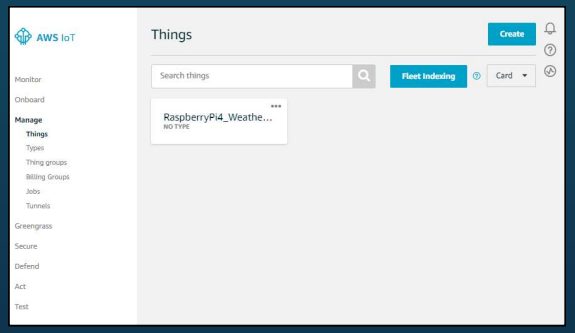


REGISTER YOUR **THING**

6. Click the “Register thing” button
   - The security policy will be created and applied in future steps

7. **Congratulations**… you now have a Thing registered with the AWS IoT Platform.





IOT **GETTING STARTED**

01

Register Your Thing

Register your device with the AWS IoT Core Framework to allow it to be managed by the platform.

02

Secure Your Thing

Secure your device by generating the required certificates and policies to allow secure communication between your gateway and the platform. In addition, we will register our device with AWS Cognito for Unauthenticated Identities.

03

Develop and Deploy Your Thing

Develop the require code to support your functionality with the AWS SDK for IoT.

04

Test Your Thing

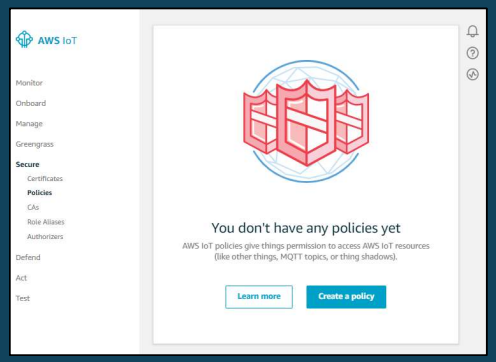
Utilize the built-in testing tools provided with the AWS IoT platform to ensure two-way communication with your device.

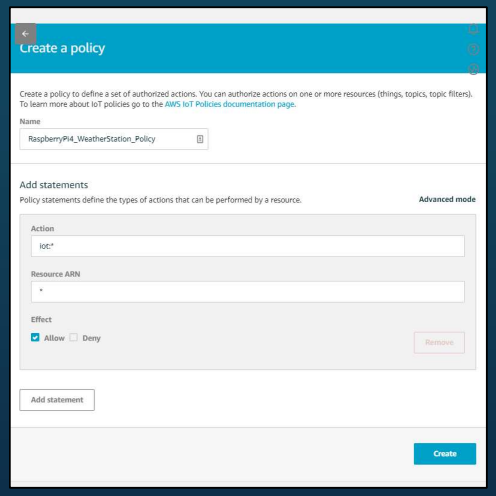
SECURE YOUR **THING**

1. Using the IoT menu, select “Secure”, then “Policies”

2. Give the policy a name, domain for actions, a resource ARN and the type of effect for the policy
   - *Name: RaspberryPi4_WeatherStation_Policy*
   - *Action: iot:**
   - *Resource ARN: **
   - *Effect: Allow checked*

Click “Create”





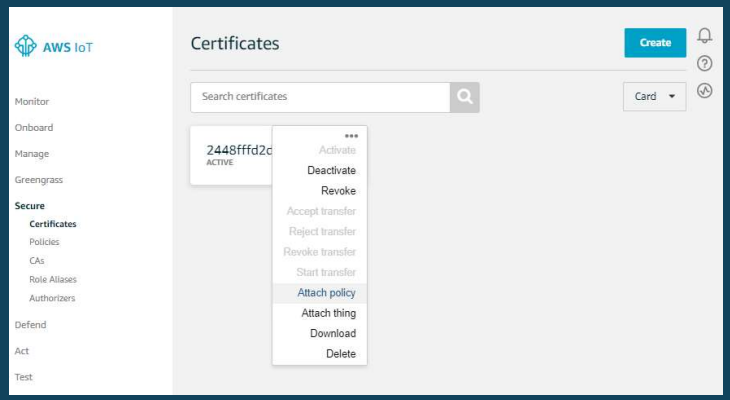
*Note: I purposely used the * in our action and ARN to generate an issue in Device Defender, so my security colleagues can lower their hands now. Normally, you may want to be more restrictive.*

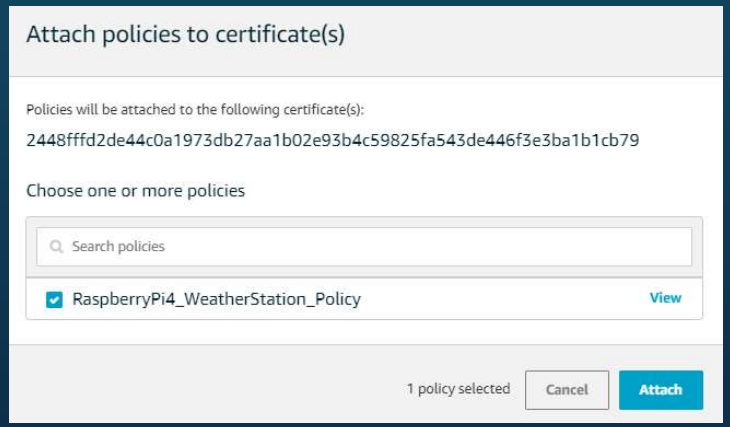
SECURE YOUR **THING**

3. Using the IoT menu, select “Secure”, then “Certificates”

4. From your Certificates action menu, select “Attach Policy”

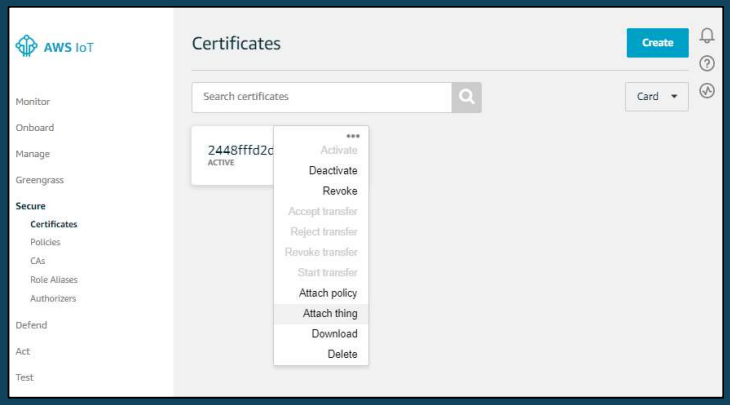
5. Check the Policy that was created and click the “Attach” button

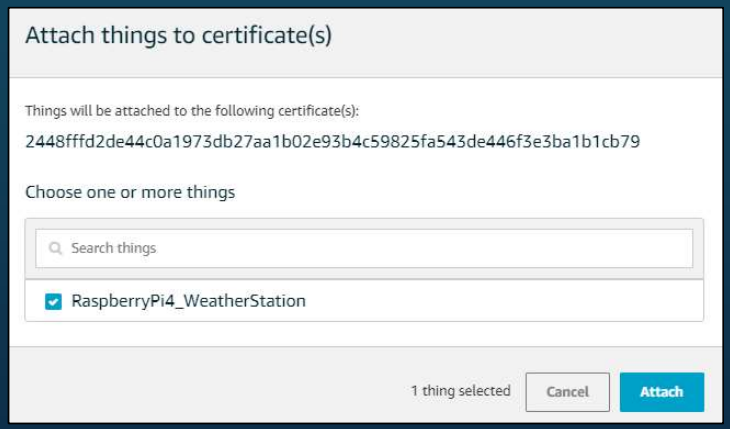




SECURE YOUR **THING**

6. Using the IoT menu, select “Secure”, then “Certificates”
7. From your Certificates action menu, select “Attach thing”
8. Check the Thing that was created and click the “Attach” button





SECURE YOUR **THING**

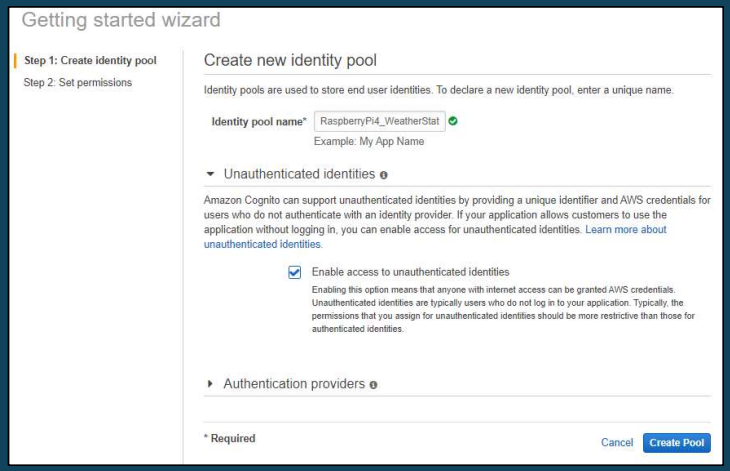
9. Using the AWS menu, search for and select “Cognito”

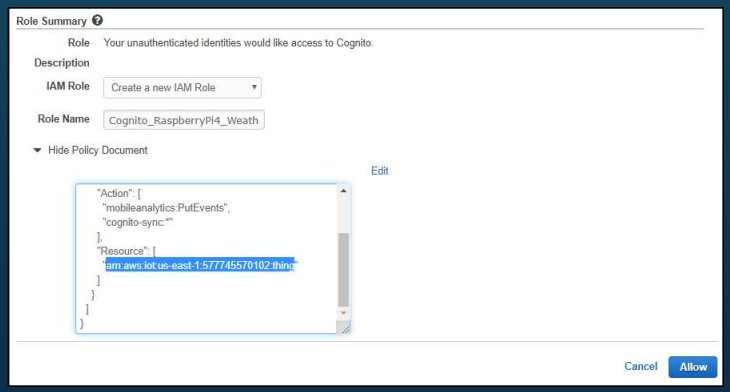
10. From the Cognito menu, select to “Manage Federated Identities”

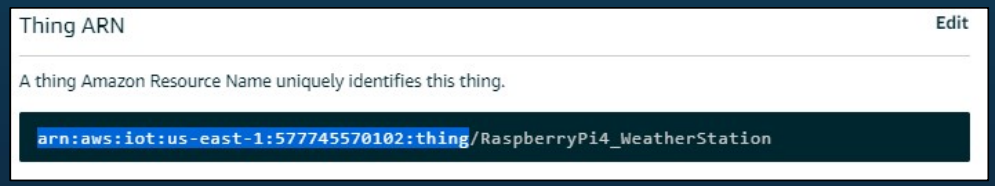
11. Give the Identity Pool a Name and Check the box to enable unauthenticated identities, click then “Create”

    Name: RaspberryPi4_WeatherStation_IdentityPool

12. Using the IoT menu, find the ARN of your Thing and enter it as the “Resource” for unauthenticated identities, then click “Allow”







IOT **GETTING STARTED**

01

Register Your Thing

Register your device with the AWS IoT Core Framework to allow it to be managed by the platform.

02

Secure Your Thing

Secure your device by generating the required certificates and policies to allow secure communication between your gateway and the platform. In addition, we will register our device with AWS Cognito for Unauthenticated Identities.

03

Develop and Deploy Your Thing

Develop the require code to support your functionality with the AWS SDK for IoT.

04

Test Your Thing

Utilize the built-in testing tools provided with the AWS IoT platform to ensure two-way communication with your device.

DEVELOP AND DEPLOY YOUR **THING**

1. Update your Raspberry Pi 4 to the latest firmware and patch levels

```
sudo rpi-update
sudo reboot now
sudo apt update
sudo apt full-upgrade
sudo apt clean
sudo reboot now
```

2. Setup our Project directory

```
sudo mkdir /var/project
sudo chgrp pi /var/project
sudo chmod 775 /var/project
```

DEVELOP AND DEPLOY YOUR **THING**

3 Install the Sense HAT Software Development Kit

```
sudo apt install sense-hat
sudo reboot
```

4 Install the AWS IoT Software Development Kit for Python

```
cd /var/project
git clone https://github.com/aws/aws-iot-device-sdk-python.git
cd aws-iot-device-sdk-python
sudo python setup.py install
```

Note: A version 2 of this Python SDK exists that is built upon the ACS (AWS Common Software).

5 Copy the 4 AWS Certificates that were generated to our projects/certs folder

DEVELOP AND DEPLOY YOUR **THING**

6 Develop the Python code to Publish/Subscribe event data with AWS IoT

- Establish our imports
- Initialize our logging
- Create Constants to manage our AWS IoT Configuration

weather-station.py

```python
#!/usr/bin/python
from AWSIoTPythonSDK.MQTTLib import AWSIoTMQTTClient
from sense_hat import SenseHat
from datetime import datetime

import time
import sys
import json
import logging

logging.basicConfig()
logger = logging.getLogger("aws_iot_weather_station")
logger.setLevel(logging.INFO)

IOT_DEVICE_CLIENT_ID = "RaspberryPi4_WeatherStation"                    #String that denotes the client identifier used to connect to AWS IoT
IOT_END_POINT_URL = "antwe8zdhpgnb-ats.iot.us-east-1.amazonaws.com"     #The host name of the user-specific AWS IoT endpoint
IOT_END_POINT_PORT = 8883                                               #Integer that denotes the port number to connect [8883/TLS1.2]
IOT_CERTIFICATE_PATH = "/var/project/certs/"                            #Used to configure the rootCA, private key and certificate file locations
IOT_CERTIFICATE_PREFIX = "863cc3ae71"                                   #Prefix of the certificate files that were created by AWS IoT
IOT_SUBSCRIBE_TOPIC = "output/weather_station"                          #The topic on which the device will receive data from the AWS IoT Platform (platform output)
IOT_PUBLISH_TOPIC = "input/weather_station"                             #The topic on which the device will send data to the AWS IoT Platform (platform input)
```

DEVELOP AND DEPLOY YOUR **THING**

6 Develop the Python code to Publish/Subscribe event data with AWS IoT (cont.)

- Define a function to manage our AWS IoT Client connection
- Define functions to manage our Publish and Subscribe functionality

weather-station.py

```python
#Configure and establish the AWS IoT Client Connection
def setup_iot_client():
    logger.info("Initializing AWS IoT Client...")
    myIoTClient = AWSIoTMQTTClient(IOT_DEVICE_CLIENT_ID)
    myIoTClient.configureEndpoint(IOT_END_POINT_URL, IOT_END_POINT_PORT)
    myIoTClient.configureCredentials(\
        "{0}root-ca.pem".format(IOT_CERTIFICATE_PATH),\
        "{0}{1}-private.pem.key".format(IOT_CERTIFICATE_PATH, IOT_CERTIFICATE_PREFIX),\
        "{0}{1}-certificate.pem.crt".format(IOT_CERTIFICATE_PATH, IOT_CERTIFICATE_PREFIX))  #Used to configure the rootCA, private key and certificate files
    myIoTClient.configureOfflinePublishQueueing(-1)                                          #Used to configure the queue size and drop behavior for the offline requests queueing (-1 is infini
    myIoTClient.configureDrainingFrequency(5)                                                #Used to configure the draining speed to clear up the queued requests when the connection is back
    myIoTClient.configureConnectDisconnectTimeout(10)                                        #Used to configure the time in seconds to wait for a disconnect to complete
    myIoTClient.configureMQTTOperationTimeout(5)                                             #Used to configure the timeout in seconds for MQTT QoS publish, subscribe and unsubscribe
    status = myIoTClient.connect()

    if status == True:
        logger.info("AWS IoT Client Connection Established...")

    return myIoTClient

#Subscribe to an AWS IoT Topic for messages
def subscribe_to_iot_topic(client, topic, function):
    return client.subscribe(topic, 0, function)

#Publish a message to an AWS IoT Topic
def publish_iot_message(client, topic, message):
    return client.publish(topic, message, 0)

#Invoked when a message is sent to this device
def receive_iot_message(client, userdata, message):
    logger.info("Topic: %s", message.topic)
    logger.info("Payload: %s", message.payload)
```

DEVELOP AND DEPLOY YOUR **THING**

6 Develop the Python code to Publish/Subscribe event data with AWS IoT (cont.)

- Define a function to manage our Sense HAT client
- Define helper functions capture and calibrate our sensor readings and define logos

weather-station.py

```python
#Initialize the Raspberry Pi SenseHat
def init_sense_hat():
    logger.info("Initializing Raspberry Pi SenseHat...")
    sense = SenseHat()
    sense.clear()
    sense.set_rotation(180)
    sense.set_pixels(get_pi_logo())
    return sense

#Convert Celcius to Fahrenheit
def convert_fahrenheit(temp):
    return 1.8 * (temp) + 32

#Return a Calibrated Temperature based upon Device Temperate minus an offset to
#account for internal thermal warning of the Raspberry Pi board
def get_calibrated_temp(sense):
    tempf = convert_fahrenheit(sense.get_temperature())
    tempcf = tempf - 38.8   #handle the thermal tempature increase from CPU heat ~ rough estimate
    return tempcf

def get_pi_logo():
    return [
    O, G, G, O, O, G, G, O,
    O, O, G, G, G, G, O, O,
    O, O, R, R, R, R, O, O,
    O, R, R, R, R, R, R, O,
    R, R, R, R, R, R, R, R,
    R, R, R, R, R, R, R, R,
    O, R, R, R, R, R, R, O,
    O, O, R, R, R, R, O, O,
    ]

def get_checkmark_logo():
    return [
    O, O, G, G, G, G, O, O,
    O, G, G, G, G, G, G, O,
    G, G, G, G, G, G, W, G,
    G, G, G, G, G, W, G, G,
    G, G, G, G, W, G, G, G,
    G, W, G, W, G, G, G, G,
    O, G, W, G, G, G, G, O,
    O, O, G, G, G, G, O, O,
    ]
```

DEVELOP AND DEPLOY YOUR **THING**

6 Develop the Python code to Publish/Subscribe event data with AWS IoT (cont.)

- Define the primary function that collects our sensor data and publishes it to AWS
- Statistics are collected every 2 seconds and averaged over 30 second periods

weather-station.py

```python
def main():
    logger.info("Starting AWS IoT Weather Station...")
    sense = init_sense_hat()
    myIoTClient = setup_iot_client()
    subscribe_to_iot_topic(myIoTClient, IOT_SUBSCRIBE_TOPIC, receive_iot_message)
    t, h = [0]*15, [0]*15
    try:
        while True:
            sense.set_pixels(get_pi_logo())
            #Measure every 2 seconds for 30 seconds, average the results and publish
            for x in range(1, 16):
                t[x-1], h[x-1] = get_calibrated_temp(sense), sense.get_humidity()

                if(x % 15 == 0):
                    temp, humidity = sum(t)/len(t), sum(h)/len(h)
                    payload = json.dumps(\
                        {"deviceId" : IOT_DEVICE_CLIENT_ID,\
                        "timestamp" : datetime.now().isoformat(),\
                        "temperature" : temp,\
                        "humidity" : humidity})
                    logger.info("Payload: %s", payload)
                    result = publish_iot_message(myIoTClient, IOT_PUBLISH_TOPIC, payload)
                    t, h = [0]*15, [0]*15
                    if(result == True):
                        sense.set_pixels(get_checkmark_logo())
                time.sleep(2)

    except KeyboardInterrupt:
        pass

    finally:
        sense.clear()
        myIoTClient.disconnect()

main()
```

JSON Payload

```
{
  deviceId: "Device Name",
  timestamp: "ISO Timestamp",
  temperature: Float,
  humidity: Float
}
```

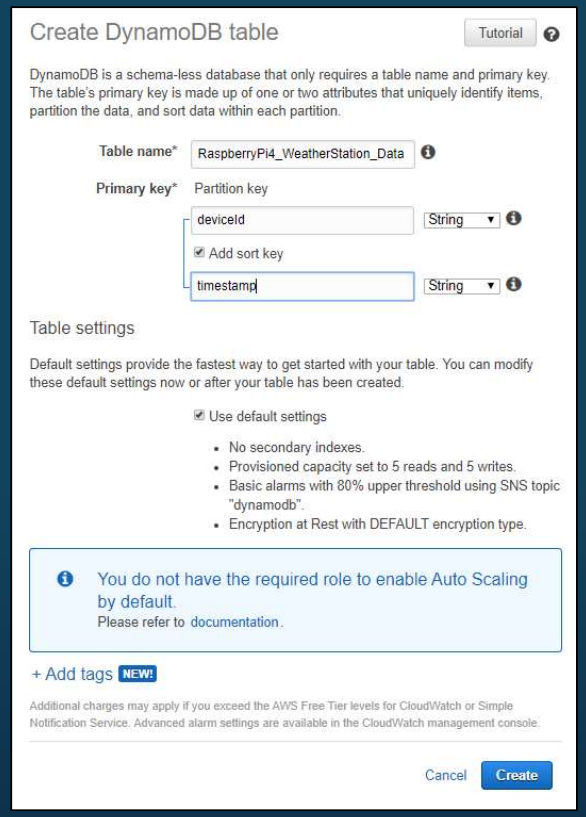
DEVELOP AND DEPLOY YOUR **THING**

7 Create a DynamoDB to store our Weather Station data

TableName: RaspberryPi4_WeatherStation_Data

Partition Key: deviceId

Sort Key: timestamp



DEVELOP AND DEPLOY YOUR **THING**

8. Create an IoT Rule to persist data to DynamoDB by selecting "Act" from the IoT Menu and clicking "Create Rule" and completing the form

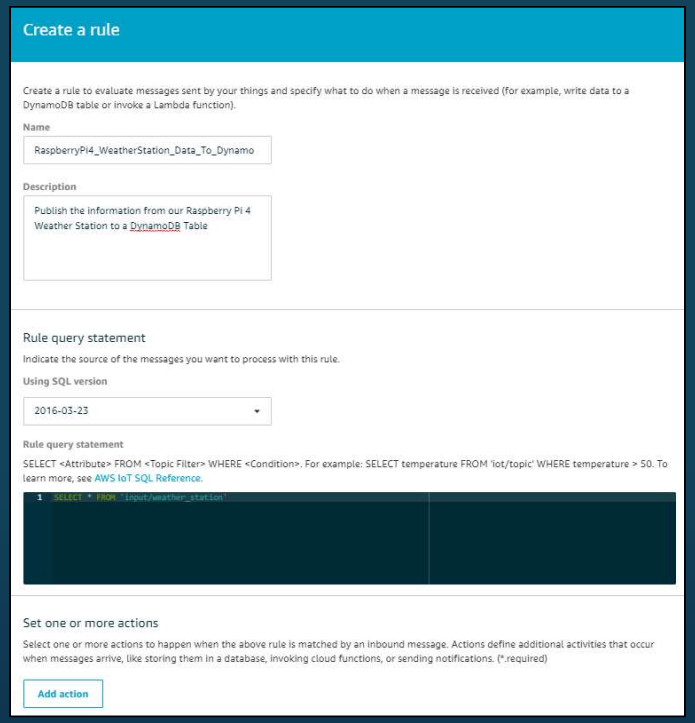
   Name: RaspberryPi4_WeatherStation_Data_To_Dynamo

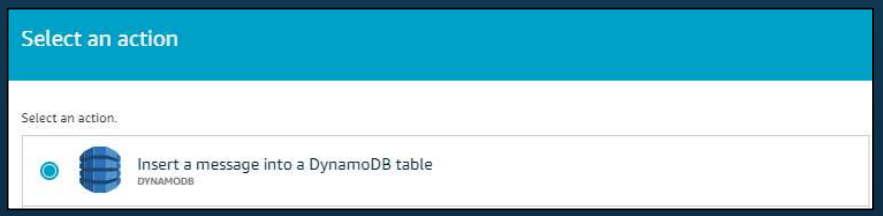
   Description: Publish the information from our Raspberry Pi 4...

   Rule Query Statement: `SELECT * FROM 'input/weather_station'`

   - *Note: '#' can be used to collect from all Topics*

9. Click the "Add action" button, select "Insert a message into a DynamoDB table, then click "Configure action"





DEVELOP AND DEPLOY YOUR **THING**

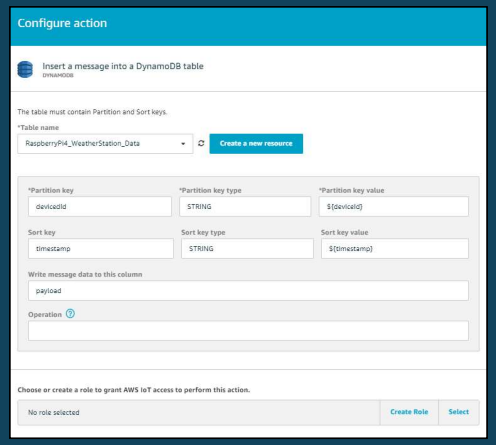
10 Complete the form with the following information”

TableName: RaspberryPi4_WeatherStation_Data

Partition key value: ${deviceId}

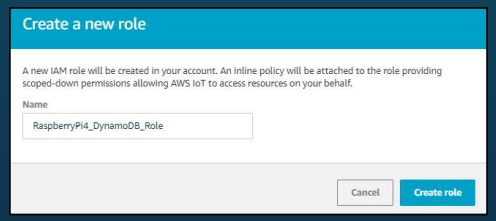
Sort key value: ${timestamp}

Write message data to this column: payload



11 Click the “Create Role” Button

Name: RaspberryPi4_DynamoDB_Role



Click “Create role” button, then Click the “Add action”, then Click the “Create rule” button

IOT **GETTING STARTED**

01

Register Your Thing

Register your device with the AWS IoT Core Framework to allow it to be managed by the platform.

02

Secure Your Thing

Secure your device by generating the required certificates and policies to allow secure communication between your gateway and the platform. In addition, we will register our device with AWS Cognito for Unauthenticated Identities.

03

Develop and Deploy Your Thing

Develop the require code to support your functionality with the AWS SDK for IoT.

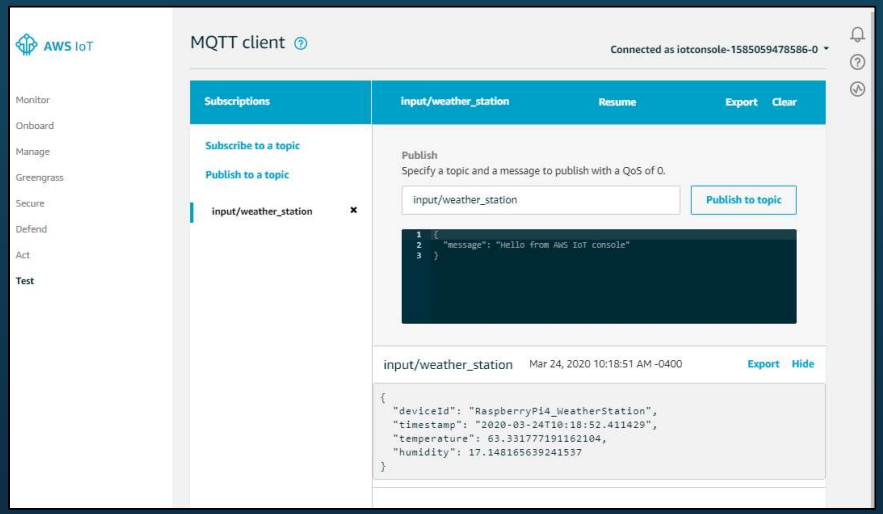
04

Test Your Thing

Utilize the built-in testing tools provided with the AWS IoT platform to ensure two-way communication with your device.

TEST YOUR **THING**

1. Using the IoT menu, select “Test”, then “Subscribe to a topic”
2. Enter the name of your input Topic to subscribe to
   Topic Name: input/weather_station
3. Confirm the JSON messages as they appear



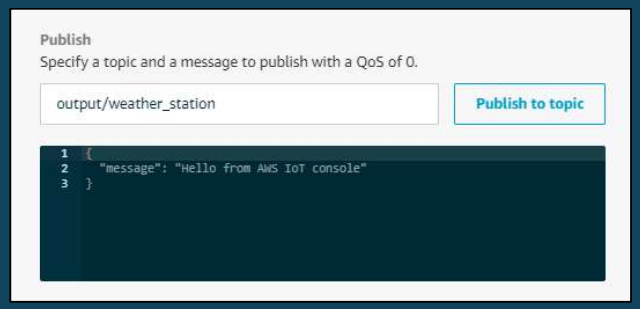
TEST YOUR **THING**

4. Using the IoT menu, select “Test”, then “Publish to a topic”

5. Enter the name of your output Topic to subscribe to

   Topic Name: output/weather_station

6. Confirm the JSON messages as they appear in your Python Console



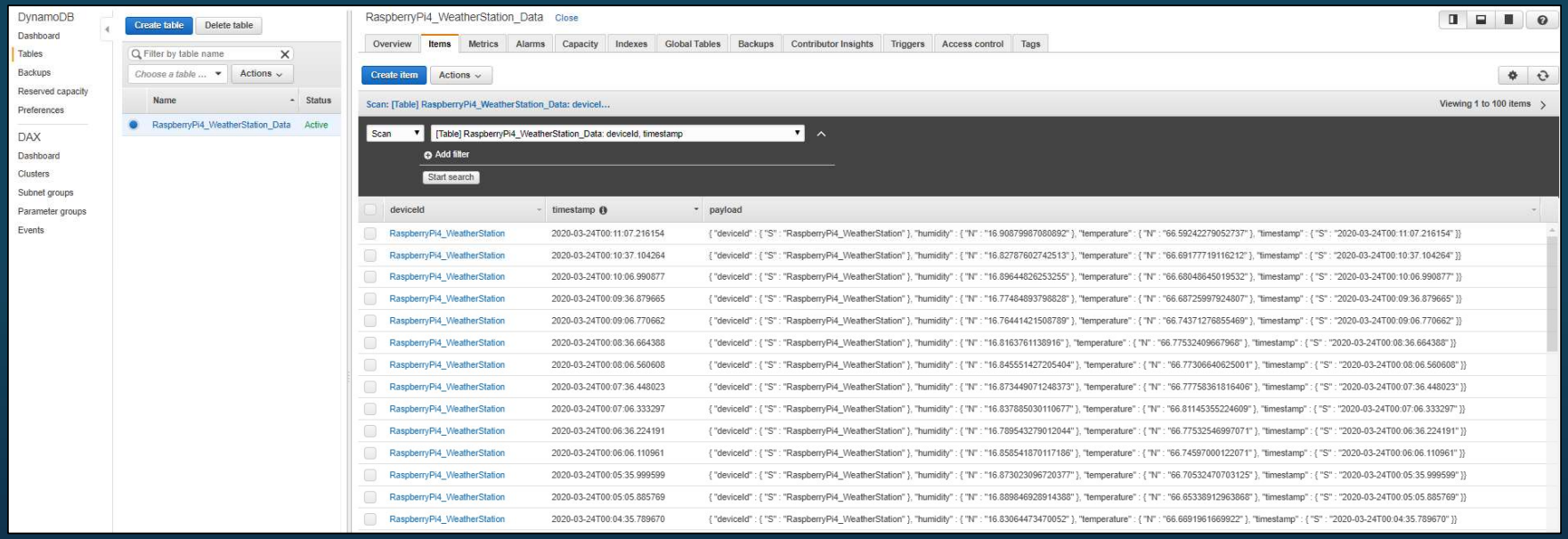

```
>>> %Run weather-station.py

 INFO:aws_iot_weather_station:Starting AWS IoT Weather Station...
 INFO:aws_iot_weather_station:Initializing Raspberry Pi SenseHat...
 INFO:aws_iot_weather_station:Initializing AWS IoT Client...
 INFO:aws_iot_weather_station:AWS IoT Client Connection Established...
 INFO:aws_iot_weather_station:Topic: output/weather_station
 INFO:aws_iot_weather_station:Payload: b'{\n  "message": "Hello from AWS IoT console"\n}'

>>>
```

TEST YOUR **THING**

7 Validate that you see the Weather Data being written to DynamoDB, by viewing the items in the table created



IOT **GETTING STARTED**

01

Register Your Thing

Register your device with the AWS IoT Core Framework to allow it to be managed by the platform.

02

Secure Your Thing

Secure your device by generating the required certificates and policies to allow secure communication between your gateway and the platform. In addition, we will register our device with AWS Cognito for Unauthenticated Identities.

03

Develop and Deploy Your Thing

Develop the require code to support your functionality with the AWS SDK for IoT.

04

Test Your Thing

Utilize the built-in testing tools provided with the AWS IoT platform to ensure two-way communication with your device.

AWS IOT **WEATHER STATION DEMO**

- Review the AWS IoT Configuration
- See the Weather Station "In Action"
- Review an IoT Device audit with Device Defender
- Resolve an issue identified by Device Defender

DEMO TIME

AWS **COST MODEL**

- The AWS pricing model is built for scale, we could add another 10 devices at virtually the same cost (*~$0.26 per device average*)

AWS Service	Usage	Estimated Monthly Cost
AWS IoT Core – Connectivity	90k calls @ 20ms/call ~ 30 minutes used	<$0.01
AWS IoT Core – Messages	Priced per million messages ~ 90k used	$1.00
AWS IoT Core – Rules Engine	Priced per million triggers/actions ~ 90k used	$0.30
AWS Device Management – Registration	Priced per 1,000 devices ~ 1 device used	$0.10
AWS Device Defender – Audit	Priced per 100,000 devices ~ 1 device used	<$0.01
Amazon Cognito	Free for under 50,000 identities	$0.00
Amazon Dynamo DB	Priced per million read/write ops ~ 90k used	$1.25
TOTAL MONTHLY COST		**$2.67**

TECHNOLOGY **LINKS**

Hardware Components

- CanaKit Raspberry Pi 4 4GB Starter MAX Kit - 64GB Edition

https://www.amazon.com/CanaKit-Raspberry-4GB-Starter-MAX/dp/B07XPHWPRB/ref=sr_1_3

- Raspberry Pi RASPBERRYPI-SENSEHAT Sense HAT

https://www.amazon.com/RASPBERRY-PI-RASPBERRYPI-SENSEHAT-Raspberry-Orientation-Temperature/dp/B014HDG74S/ref=sr_1_3

- Hologram Nova Global Cellular Modem

https://hologram.io/store/nova-global-cellular-modem/55

Technology

- AWS IoT Platform

https://aws.amazon.com/iot/

- AWS IoT Python SDK Documentation

https://s3.amazonaws.com/aws-iot-device-sdk-python-docs/html/index.html

- Raspberry Pi Platform

https://www.raspberrypi.org/

- Sense HAT Python SDK

https://pythonhosted.org/sense-hat/

Source Code

- AWS IoT Weather Station Python Source Code

https://github.com/orgs/cleanslate-technology-group/aws-iot-weather-station

EDUCATIONAL **LINKS**

PluralSight

- AWS IoT: The Big Picture

https://www.pluralsight.com/courses/aws-iot-big-picture

- Managing Connected Devices with AWS IoT Device Management

https://www.pluralsight.com/courses/aws-iot-device-management

- Analyzing Your Data with AWS IoT Analytics

https://www.pluralsight.com/courses/aws-iot-analytics

Linux Academy

- Configuring Your First IoT Device Lab

https://linuxacademy.com/hands-on-lab/101e6431-b174-464d-b130-527914d768d3/

A Cloud Guru

- How to Turn on a Lamp from Anywhere in the World using AWS IoT Greengrass

https://acloud.guru/series/acg-projects/view/112

LET'S **CONNECT**

CleanSlate

CleanSlate Technology Group is an AWS Advanced Partner specializing in Application Modernization using Cloud-based technologies.

Contact us to help you on your IoT Journey!

- Be Creative – think of the next must-have IoT concept and build it yourself!
- Get Ready – use the educational links in this presentation and grow your skills
- Start Simple – implement a simple PoC, like a Weather Station or Health Device
- Leverage a Platform – AWS IoT is a great platform to learn and use
- Check Out My Project - https://github.com/orgs/cleanslate-technology-group/aws-iot-weather-station

Darren Mills
darren.mills@cleanslatetg.com

THANK **YOU**

Chris Konow
chris.konow@cleanslatetg.com

OUR OFFER TO **YOU**

CleanSlate

As a THANK YOU for attending one of our speaking sessions, connect with us on our Slack Channel for one of the offers below.

#sponsor-cleanslate

Cloud Optimization Assessment with CloudCheckr - This evaluation will provide key recommendations for cloud cost optimization that will result in immediate savings opportunities – **up to 30% cost savings**

Cloud Strategic Planning Workshop – This workshop will introduce our proven Cloud Strategic Planning Framework and highlight the steps necessary to achieve successful cloud adoption including implementation of Cloud Architecture, Security, and Dev/Ops