Connected Products using AWS

Mike Vartanian

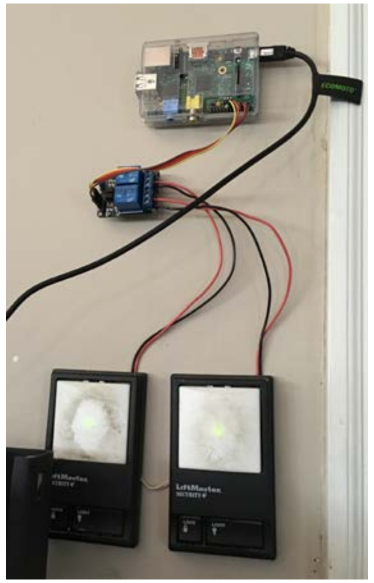
9/19/2016

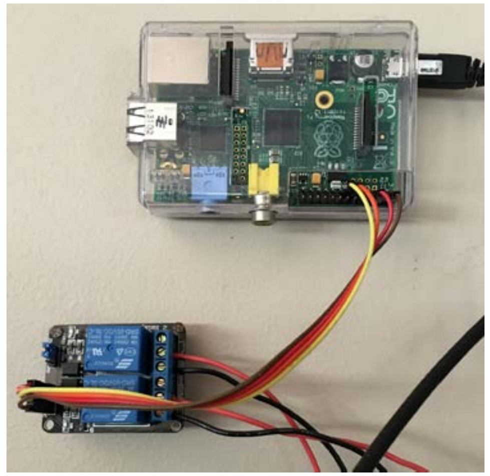
Agenda

- Problem
- System Overview
- What is MQTT?
- Specific System Building Blocks
    - Garage Door Opener / Raspberry Pi Setup
    - AWS IoT Setup
    - iOS Mobile Application
- Next Steps / Help?

Problem / Why did I use AWS to connect a product?

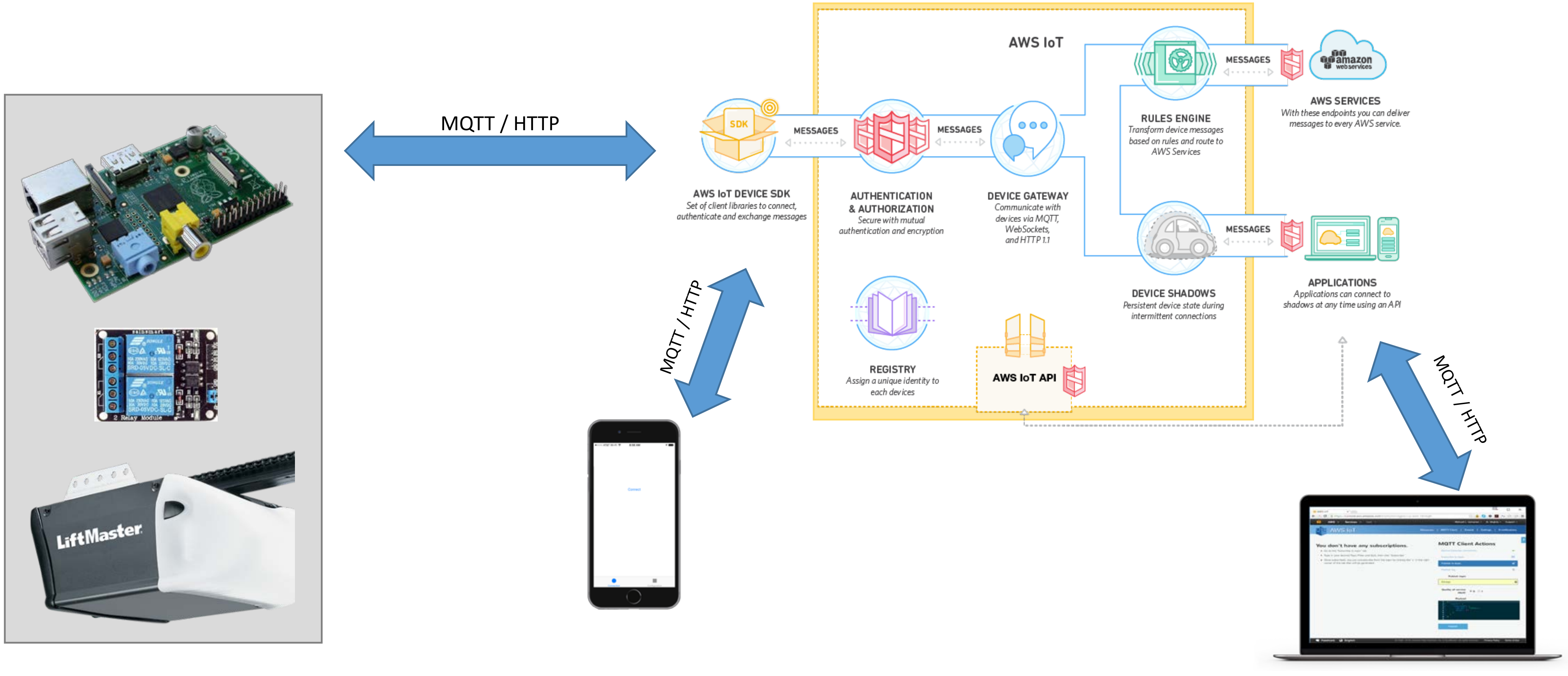
- Created connected Garage Door Opener that operated over home WiFi Network
- Served a local website on the Raspberry Pi
  - Simple HTML and Python Script using Webiopi
  - https://github.com/mvartani76/RPi-GarageDoorOpener





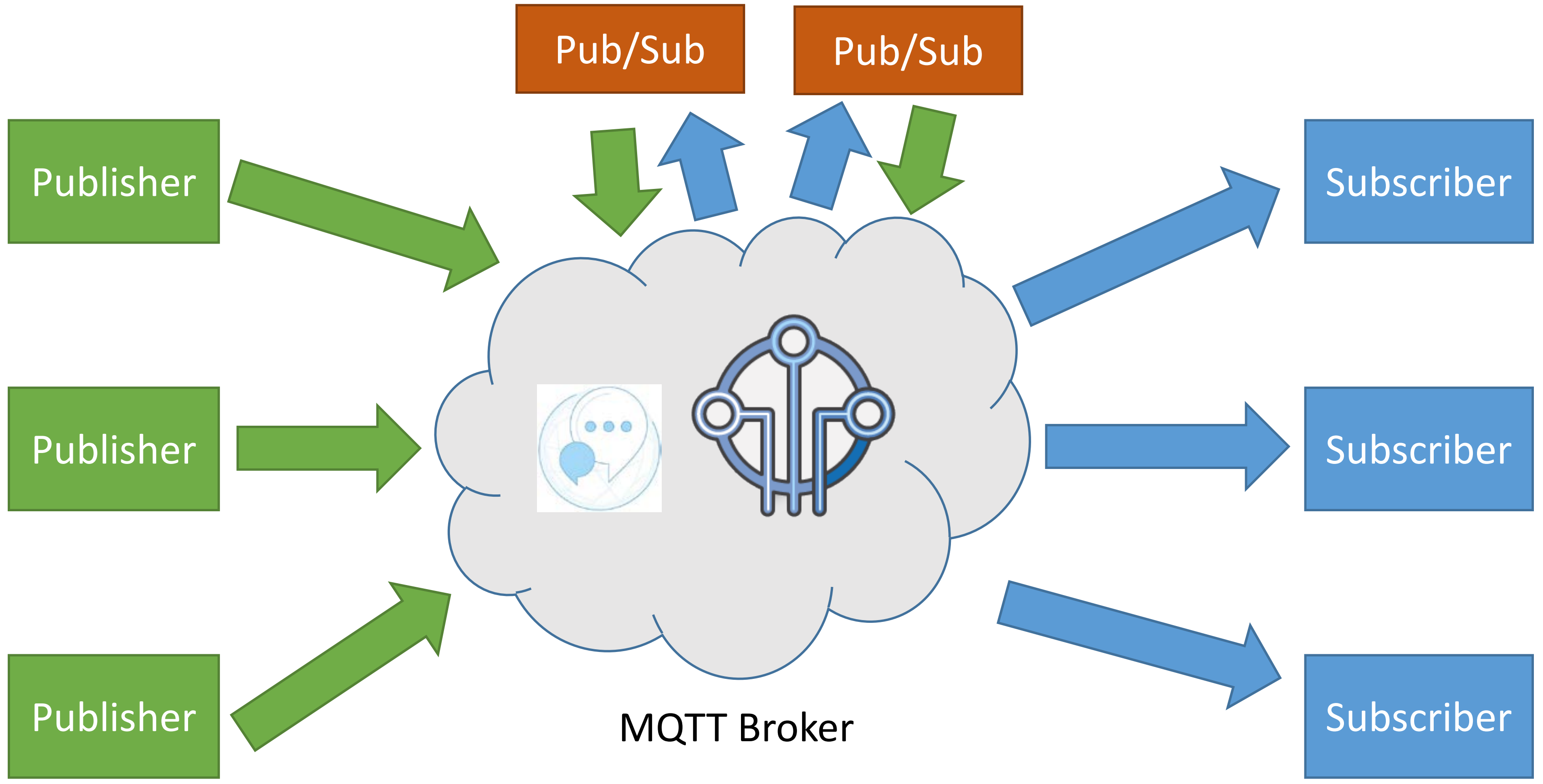
- However, could not connect remotely (or over external networks)…
  - This was okay (not great) for opening/closing when in proximity of my house
- Could not monitor status when away from my house though
  - Previous solution was to use port forwarding on my WiFi Router
  - Concerned about security and opening my network to the world
  - I could have probably set up a VPN or done something else but I am not that smart…☺

System Overview


MQTT / HTTP
LiftMaster
2 Relay Module
AWS IoT
SDK
AWS IoT DEVICE SDK
Set of client libraries to connect, authenticate and exchange messages
MESSAGES
AUTHENTICATION & AUTHORIZATION
Secure with mutual authentication and encryption
MESSAGES
DEVICE GATEWAY
Communicate with devices via MQTT, WebSockets, and HTTP 1.1
RULES ENGINE
Transform device messages based on rules and route to AWS Services
MESSAGES
amazon web services
AWS SERVICES
With these endpoints you can deliver messages to every AWS service.
DEVICE SHADOWS
Persistent device state during intermittent connections
MESSAGES
APPLICATIONS
Applications can connect to shadows at any time using an API
REGISTRY
Assign a unique identity to each devices
AWS IoT API
MQTT / HTTP
MQTT / HTTP
AWS IoT
MQTT Client Actions
You don't have any subscriptions.

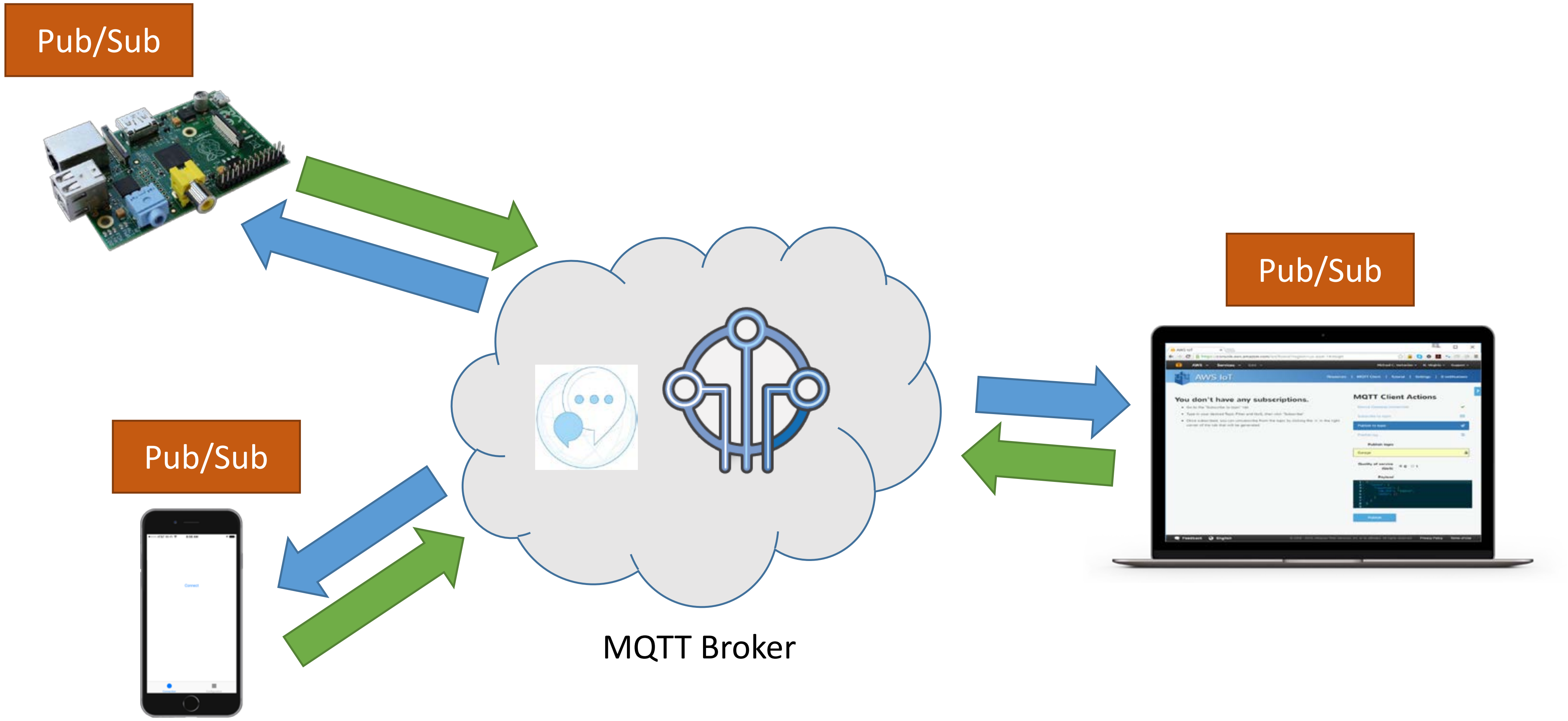
What is MQTT?

- **<u>MQ Telemetry Transport (MQTT)</u>** is a machine-to-machine (M2M) / "Internet of Things" connectivity protocol
- Designed as an extremely lightweight publish/subscribe messaging transport
- Useful for connections with remote locations where a small code footprint is required and/or network bandwidth is at a premium



MQTT Configuration/Setup for my Project

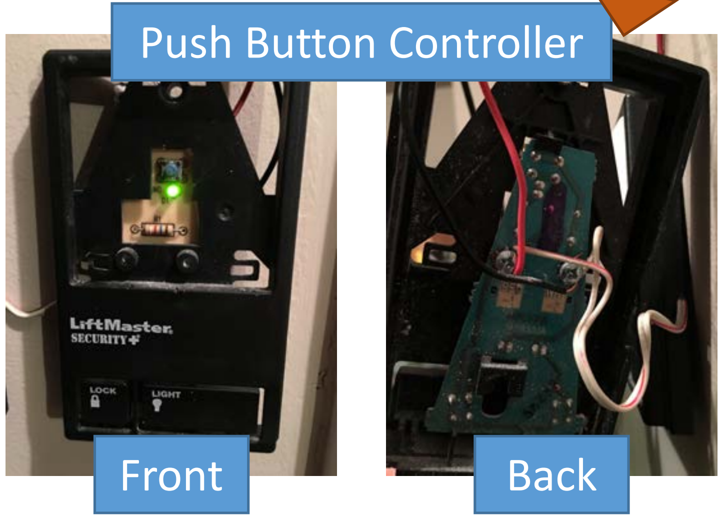
- All devices in my project operate as both a Publisher and a Subscriber and have bi-directional communication with each other through the MQTT Broker, AWS IoT

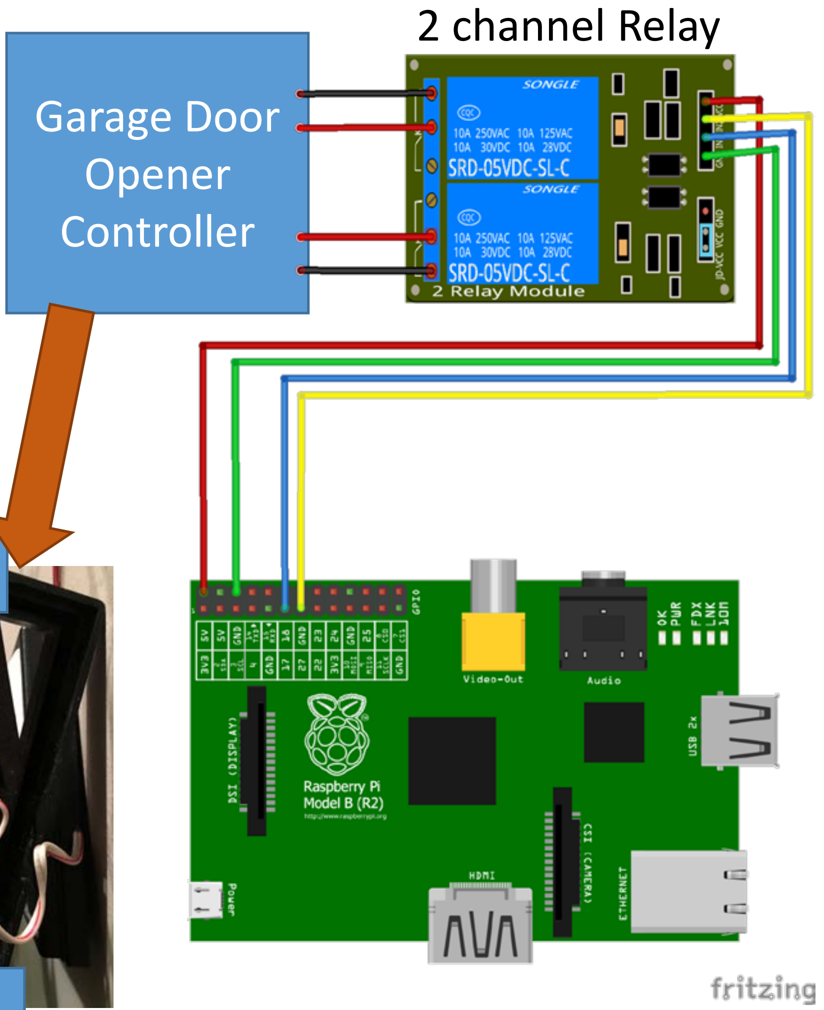


Garage Door Opener / Raspberry Pi Setup

- Connect Rpi to Garage Door Opener Controller via Relay
  - Relay isolates the Rpi from Garage Door Opener
  - https://www.amazon.com/SainSmart-101-70-100-2-Channel-Relay-Module/dp/B0057OC6D8/ref=sr_1_3?ie=UTF8&qid=1474072892&sr=8-3
  - 2 channel Relay controls 2 garage doors

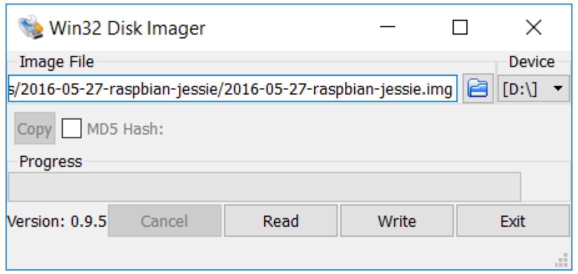
My garage door is triggered essentially by shorting the controller in a set sequence





Pre-Configure Raspberry Pi

- Using Raspbian Jessie Image (https://www.raspberrypi.org/downloads/raspbian/)
  - Used Win32DiskManager to write image to SD Card
  - Python 2.7.9 (python), Python 3.4.2 (python3), git comes pre installed



Win32 Disk Imager

Image File: s/2016-05-27-raspbian-jessie/2016-05-27-raspbian-jessie.img

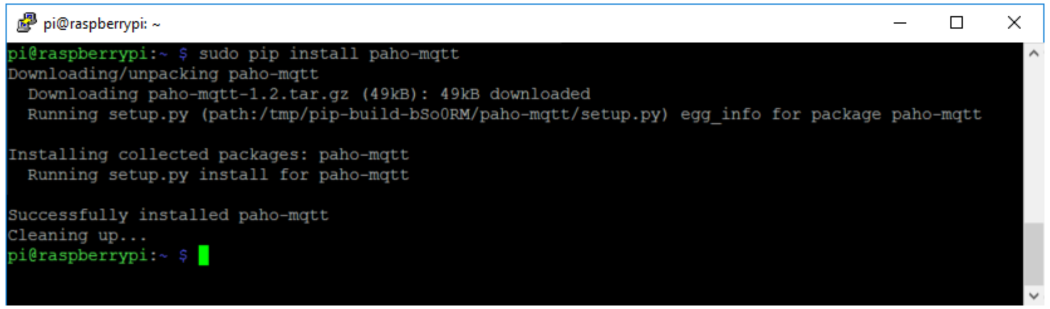
Device: [D:\]

☐ MD5 Hash:

Progress

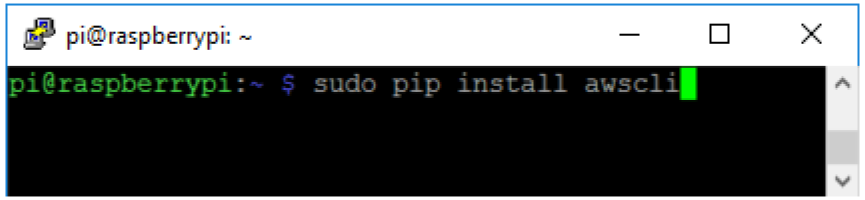
Version: 0.9.5

- Install paho-mqtt



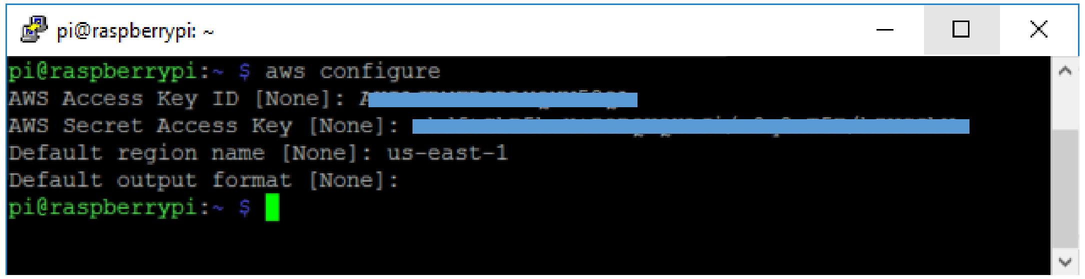
```
pi@raspberrypi:~ $ sudo pip install paho-mqtt
Downloading/unpacking paho-mqtt
  Downloading paho-mqtt-1.2.tar.gz (49kB): 49kB downloaded
  Running setup.py (path:/tmp/pip-build-bSo0RM/paho-mqtt/setup.py) egg_info for package paho-mqtt

Installing collected packages: paho-mqtt
  Running setup.py install for paho-mqtt

Successfully installed paho-mqtt
Cleaning up...
pi@raspberrypi:~ $
```

- Install awscli



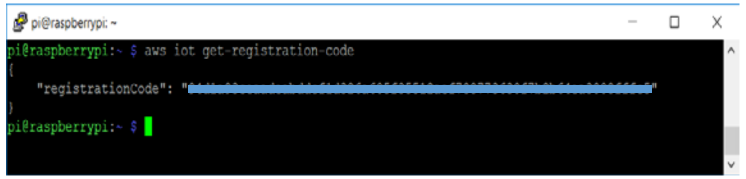
```
pi@raspberrypi:~ $ sudo pip install awscli
```

- Configure awscli



```
pi@raspberrypi:~ $ aws configure
AWS Access Key ID [None]: A[redacted]
AWS Secret Access Key [None]: [redacted]
Default region name [None]: us-east-1
Default output format [None]:
pi@raspberrypi:~ $
```

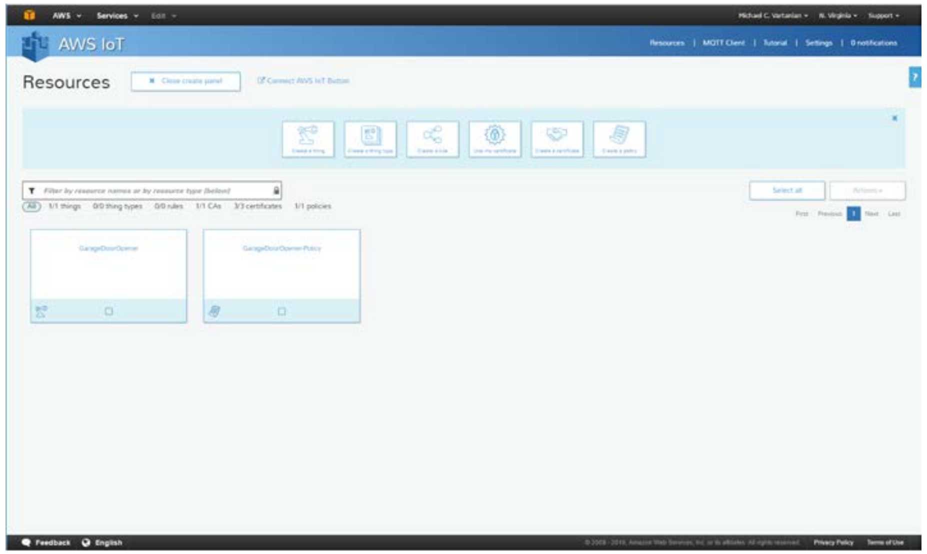
Confirm by running aws iot command

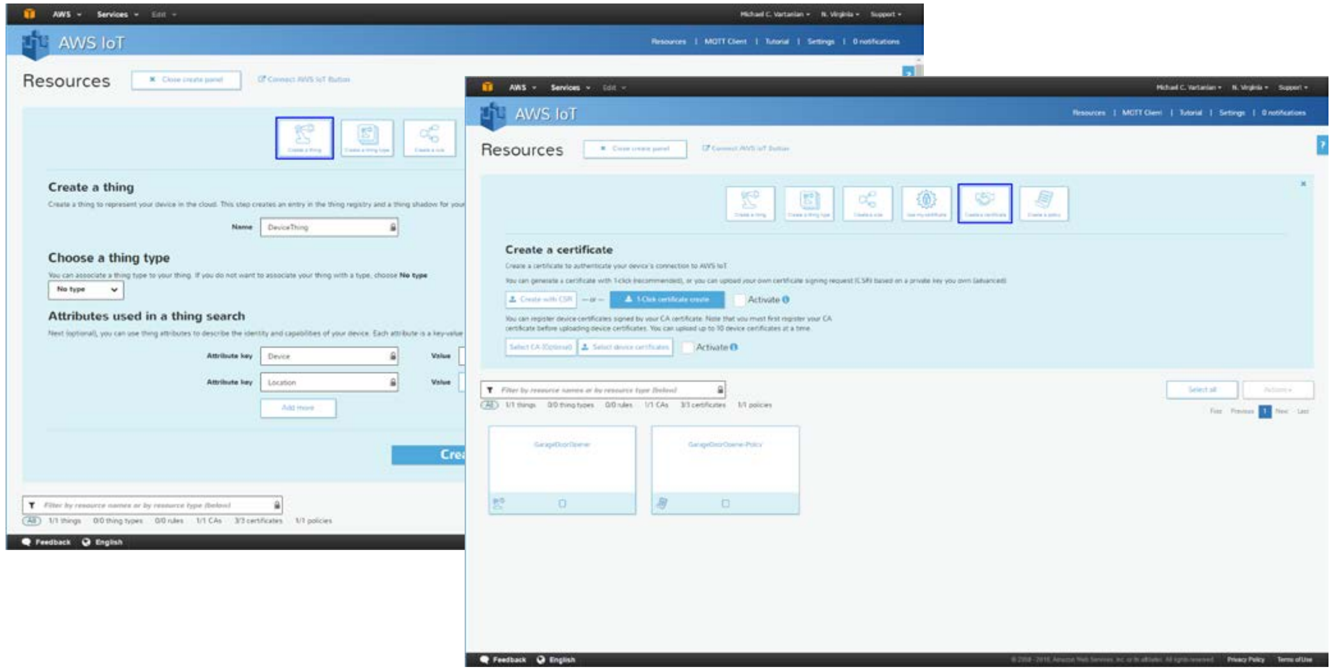


```
pi@raspberrypi:~ $ aws iot get-registration-code
{
    "registrationCode": "[redacted]"
}
pi@raspberrypi:~ $
```

Configure AWS IoT Settings

- Multiple methods to configure AWS IoT Settings
  - AWS Command Line Interface (CLI)
    - http://docs.aws.amazon.com/iot/latest/developerguide/what-is-aws-iot.html
    - http://docs.aws.amazon.com/cli/latest/userguide/cli-chap-welcome.html
      - http://docs.aws.amazon.com/cli/latest/reference/iot/index.html
  - AWS IoT Web Portal



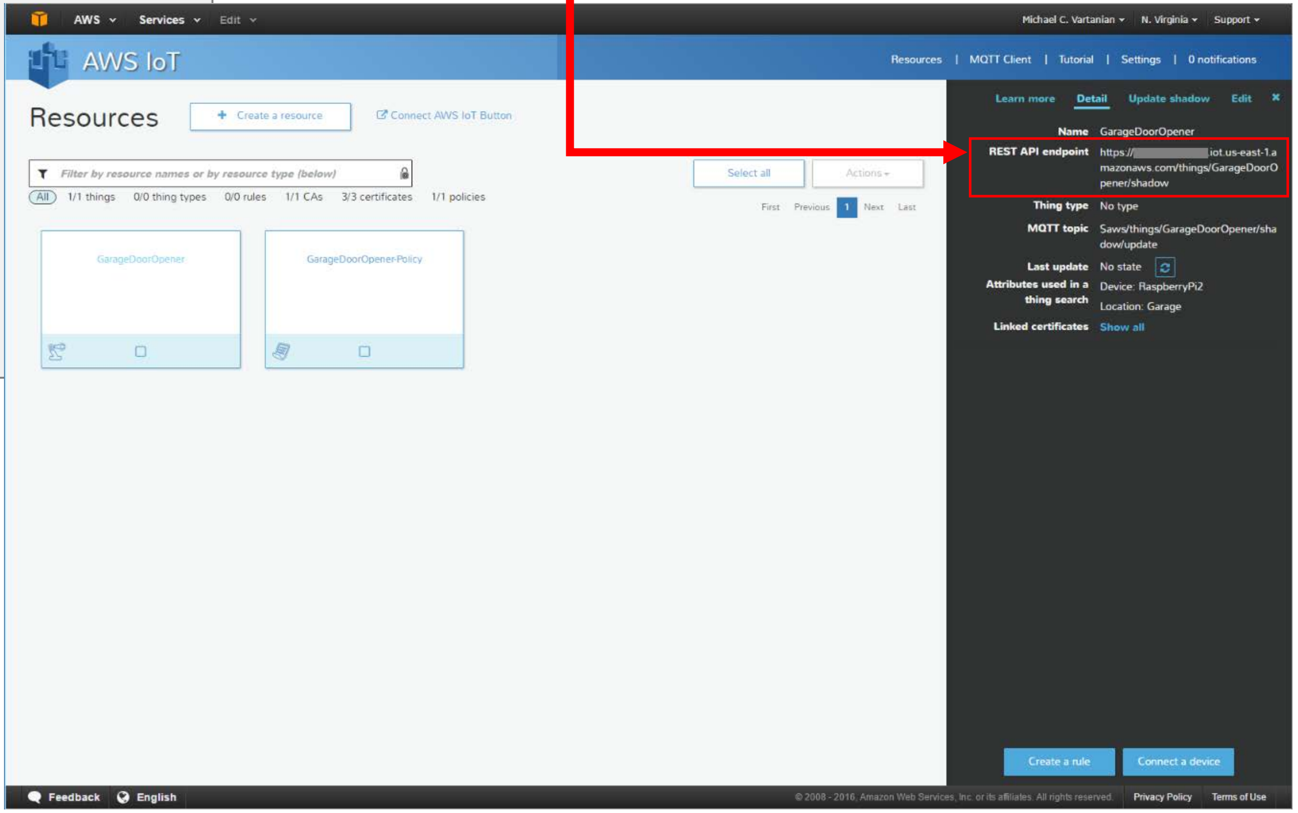


Configure AWS IoT Settings in Python

awsiot_garage.py

```
awshost = "data.iot.us-east-1.amazonaws.com"
awsport = 8883
clientId = "GarageDoorOpener"
thingName = "GarageDoorOpener"

caPath = "aws-iot-rootCA.crt"
certPath = "cert.pem"
keyPath = "privkey.pem"
```

- Follow instructions from https://github.com/mariocannistra/python-paho-mqtt-for-aws-iot to setup the keys/certificates on the Raspberry Pi
- Use `aws iot create-keys-and-certificate` AWS CLI command and Symantec Root Certificate for AWS

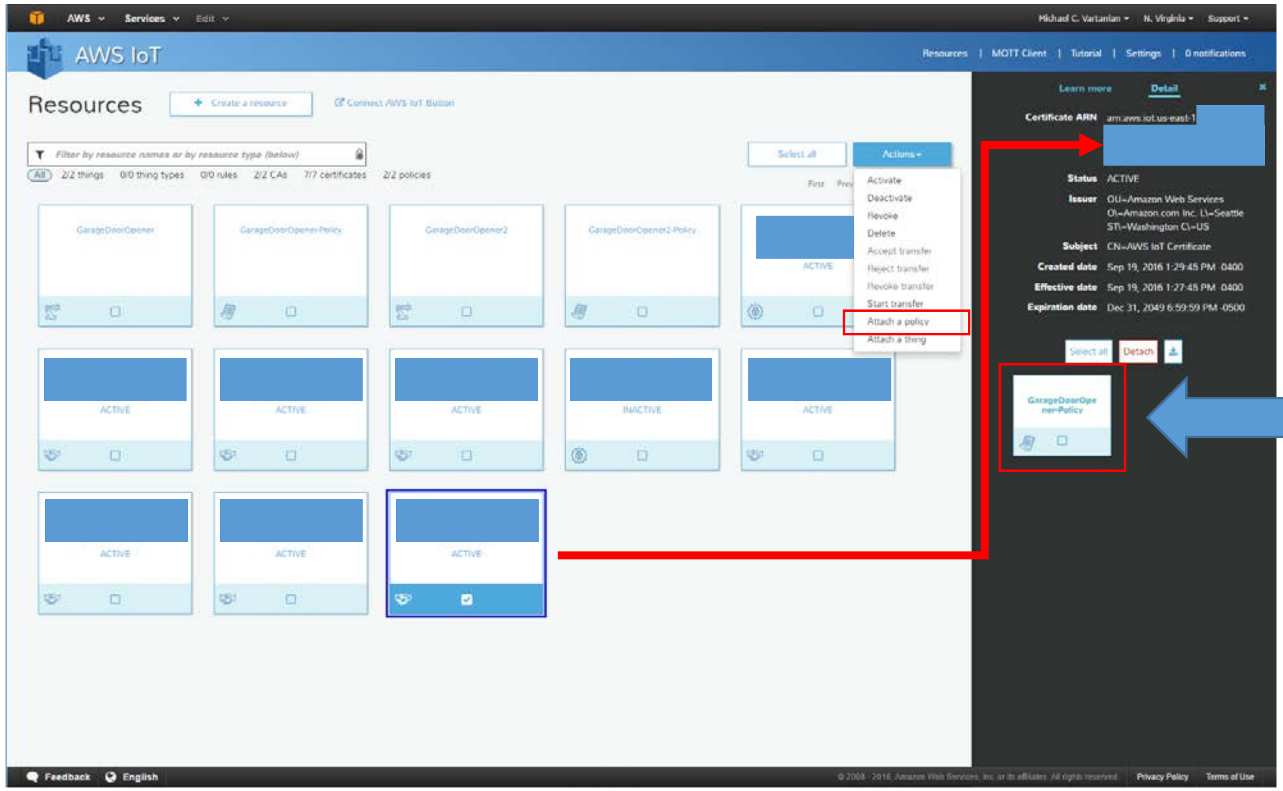


Name	GarageDoorOpener
REST API endpoint	https://[redacted]iot.us-east-1.amazonaws.com/things/GarageDoorOpener/shadow
Thing type	No type
MQTT topic	$aws/things/GarageDoorOpener/shadow/update
Last update	No state
Attributes used in a thing search	Device: RaspberryPi2 Location: Garage
Linked certificates	Show all

© 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Configure AWS IoT Settings in Python

- Generated certificate from previous `aws iot create-keys-and-certificate` AWS command still needs to be attached to a policy for the Raspberry Pi to communicate with AWS IoT



Certificate ARN arn:aws:iot:us-east-1[redacted]
Status ACTIVE
Issuer OU=Amazon Web Services O\=Amazon.com Inc. L\=Seattle ST\=Washington C\=US
Subject CN=AWS IoT Certificate
Created date Sep 19, 2016 1:29:45 PM -0400
Effective date Sep 19, 2016 1:27:45 PM -0400
Expiration date Dec 31, 2049 6:59:59 PM -0500

This confirms that certificate is attached to a policy

Python Code

- Using Paho Python Client
  - https://eclipse.org/paho/clients/python/
- Using Rpi GPIO library
  - https://pypi.python.org/pypi/RPi.GPIO
  - https://sourceforge.net/p/raspberry-gpio-python/wiki/install/
  - https://sourceforge.net/p/raspberry-gpio-python/wiki/Examples/

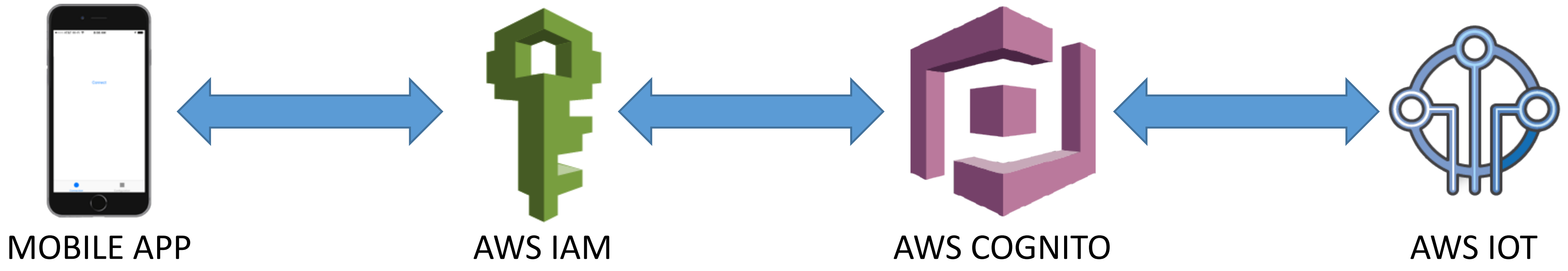
Python Code (subscribe function – on_message())

- Conditional logic inside on_message to check for Garage topic
- Not sure if this is the correct use of MQTT because I thought devices subscribed to topics and hence would only see subscribed topics?

```
def on_message(client, userdata, msg):
        print("topic: "+msg.topic)
        print("payload: "+str(msg.payload))
        if msg.topic == "Garage":
                json_msg = json.loads(msg.payload.decode())
                print json_msg["state"]["reported"]["ON_OFF"]
                if json_msg["state"]["reported"]["ON_OFF"] == "ON":
                        print "GPIO HIGH"
                        GPIO.output(json_msg["state"]["reported"]["GPIO"],GPIO.HIGH)
                elif json_msg["state"]["reported"]["ON_OFF"] == "OFF":
                        print "GPIO LOW"
                        GPIO.output(json_msg["state"]["reported"]["GPIO"],GPIO.LOW)
                elif json_msg["state"]["reported"]["ON_OFF"] == "TOGGLE":
                        GPIO.output(json_msg["state"]["reported"]["GPIO"],GPIO.LOW)
                        time.sleep(0.5)
                        GPIO.output(json_msg["state"]["reported"]["GPIO"],GPIO.HIGH)
                        time.sleep(0.5)
                        GPIO.output(json_msg["state"]["reported"]["GPIO"],GPIO.LOW)
                        time.sleep(0.5)
                        GPIO.output(json_msg["state"]["reported"]["GPIO"],GPIO.HIGH)
```

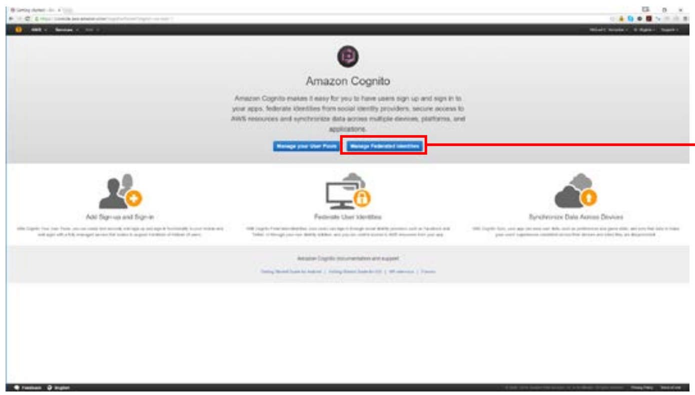
iOS Mobile Application

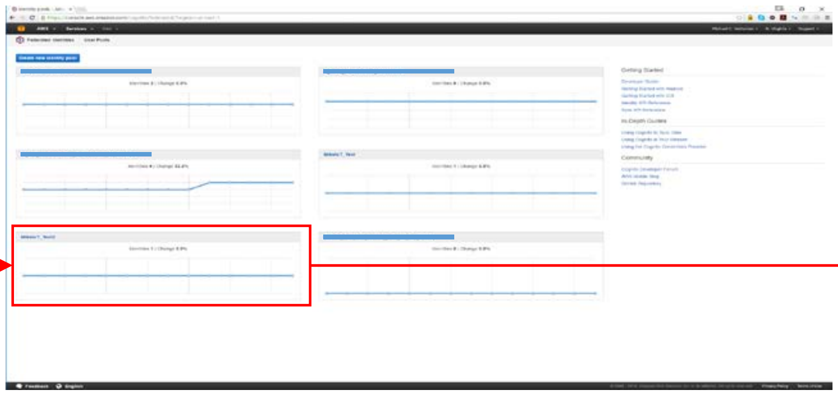
- iOS Mobile Application based on AWS IoT Sample code
  - https://github.com/awslabs/aws-sdk-ios-samples/tree/master/IoT-Sample/Swift/IoTSampleSwift
  - Application connects as a **Unauthenticated Role** using Cognito / User Pools and then creates its own Certificates/Keys for appropriate credentials
    - **Do not need to create keys a priori for this iOS Mobile App!!**
    - You can but would need to store the keys in the iOS keychain (I didn't do this)

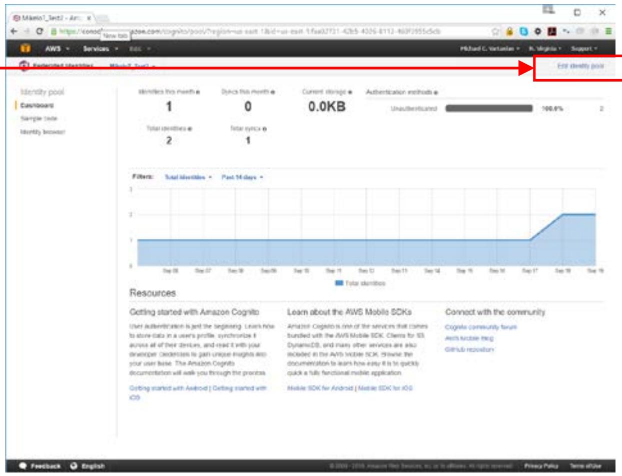


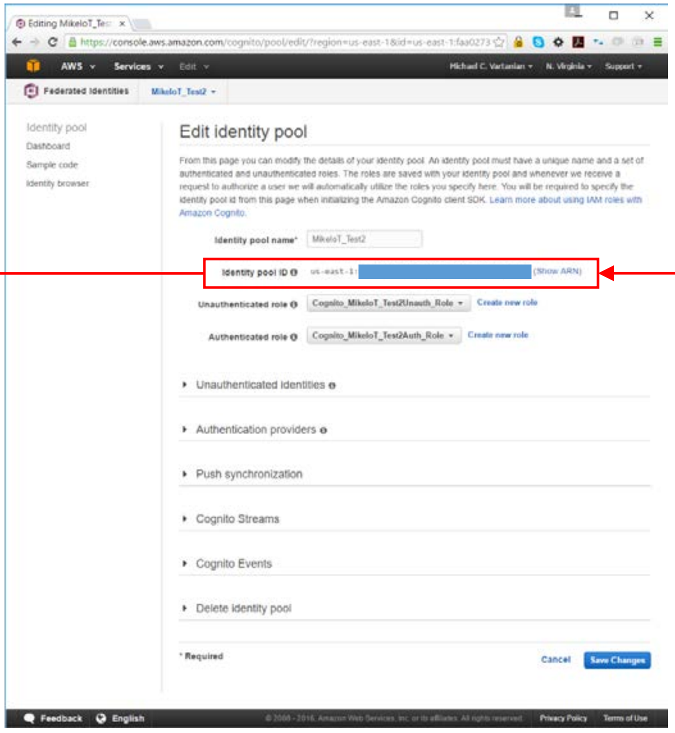
iOS Mobile Application – Cognito Pool

- Authenticate User Identity via AWS Cognito
- Configure CognitoIdentityPoolId using AWS Web Portal









Editing MikeloT_Te
https://console.aws.amazon.com/cognito/pool/edit/?region=us-east-1&id=us-east-1:faa0273

AWS · Services · Edit · Michael C. Vartanian · N. Virginia · Support

Federated Identities · MikeloT_Test2

Identity pool
Dashboard
Sample code
Identity browser

Edit identity pool

From this page you can modify the details of your identity pool. An identity pool must have a unique name and a set of authenticated and unauthenticated roles. The roles are saved with your identity pool and whenever we receive a request to authorize a user we will automatically utilize the roles you specify here. You will be required to specify the identity pool id from this page when initializing the Amazon Cognito client SDK. Learn more about using IAM roles with Amazon Cognito.

Identity pool name*: MikeloT_Test2
Identity pool ID: us-east-1:[redacted] (Show ARN)
Unauthenticated role: Cognito_MikeloT_Test2Unauth_Role · Create new role
Authenticated role: Cognito_MikeloT_Test2Auth_Role · Create new role

- Unauthenticated identities
- Authentication providers
- Push synchronization
- Cognito Streams
- Cognito Events
- Delete identity pool

* Required · Cancel · Save Changes

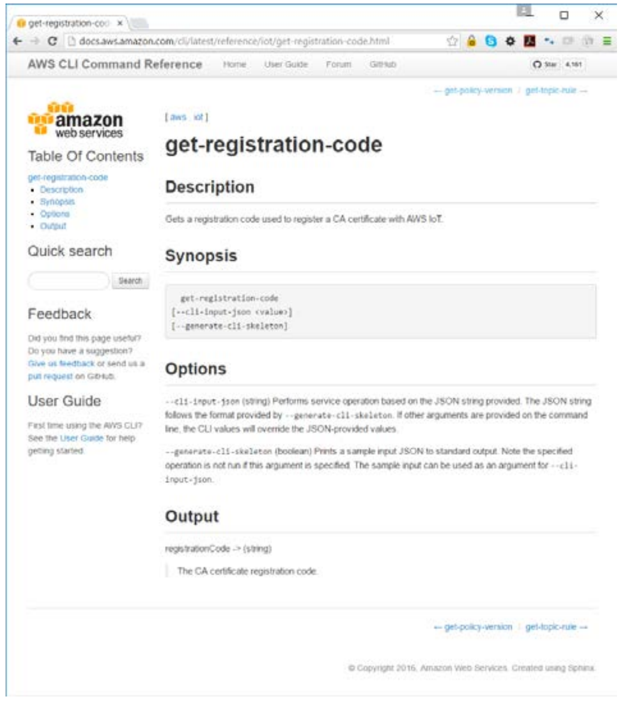
Feedback · English · © 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved. · Privacy Policy · Terms of Use

Constants.swift

```
import Foundation
import AWSCore

//WARNING: To run this sample correctly, you must set the following constants.
let AwsRegion = AWSRegionType.Unknown // e.g. AWSRegionType.USEast1
let CognitoIdentityPoolId = "YourCognitoIdentityPoolId"
let CertificateSigningRequestCommonName = "IoTSampleSwift Application"
let CertificateSigningRequestCountryName = "Your Country"
let CertificateSigningRequestOrganizationName = "Your Organization"
let CertificateSigningRequestOrganizationalUnitName = "Your Organizational Unit"
let PolicyName = "YourPolicyName"
```

iOS Mobile Application – Set Up Common Name

- iOS Mobile App uses “Common Name” or “Registration Code” to set up Certificates/Keys



AWS CLI Command Reference

get-registration-cod
docs.aws.amazon.com/cli/latest/reference/iot/get-registration-code.html
Home
User Guide
Forum
GitHub
Star 4,161
← get-policy-version | get-topic-rule →
amazon web services

Table Of Contents

get-registration-code
- Description
- Synopsis
- Options
- Output

Quick search

Search

Feedback

Did you find this page useful? Do you have a suggestion? Give us feedback or send us a pull request on GitHub.

User Guide

First time using the AWS CLI? See the User Guide for help getting started.

[aws . iot]

get-registration-code

Description

Gets a registration code used to register a CA certificate with AWS IoT.

Synopsis

```
  get-registration-code
[--cli-input-json <value>]
[--generate-cli-skeleton]
```

Options

`--cli-input-json` (string) Performs service operation based on the JSON string provided. The JSON string follows the format provided by `--generate-cli-skeleton`. If other arguments are provided on the command line, the CLI values will override the JSON-provided values.

`--generate-cli-skeleton` (boolean) Prints a sample input JSON to standard output. Note the specified operation is not run if this argument is specified. The sample input can be used as an argument for `--cli-input-json`.

Output

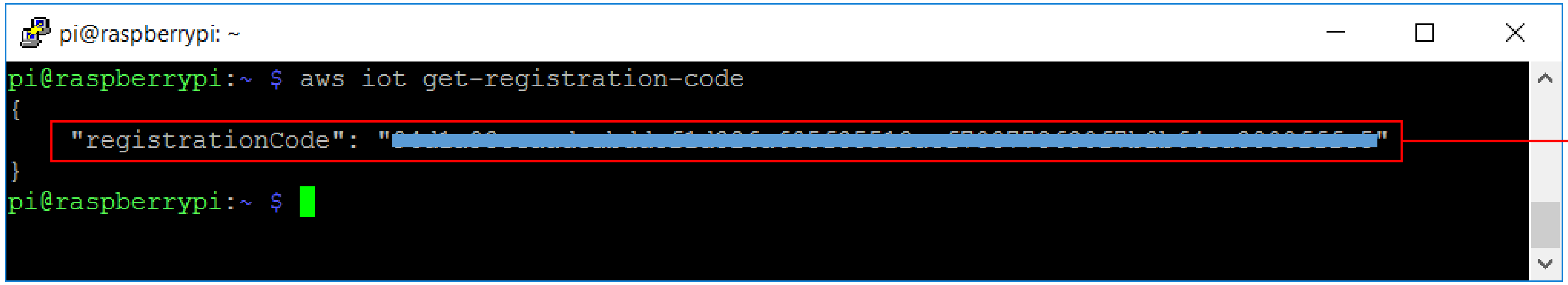
registrationCode -> (string)

The CA certificate registration code.

← get-policy-version | get-topic-rule →

© Copyright 2016, Amazon Web Services. Created using Sphinx.

SSH into Pi using AWS CLI



```
pi@raspberrypi:~ $ aws iot get-registration-code
{
    "registrationCode": "[redacted]"
}
pi@raspberrypi:~ $
```

Constants.swift

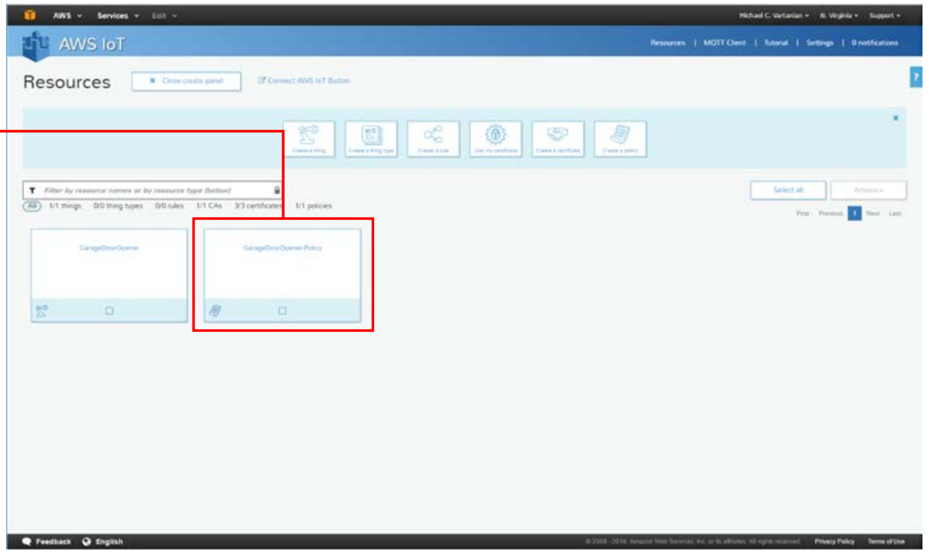
```
import Foundation
import AWSCore

//WARNING: To run this sample correctly, you must set the following constants.
let AwsRegion = AWSRegionType.Unknown // e.g. AWSRegionType.USEast1
let CognitoIdentityPoolId = "YourCognitoIdentityPoolId"
let CertificateSigningRequestCommonName = "IoTSampleSwift Application"
let CertificateSigningRequestCountryName = "Your Country"
let CertificateSigningRequestOrganizationName = "Your Organization"
let CertificateSigningRequestOrganizationalUnitName = "Your Organizational Unit"
let PolicyName = "YourPolicyName"
```

iOS Mobile Application – Policy Name

- Need Policy Name for authentication and resource information/configuration

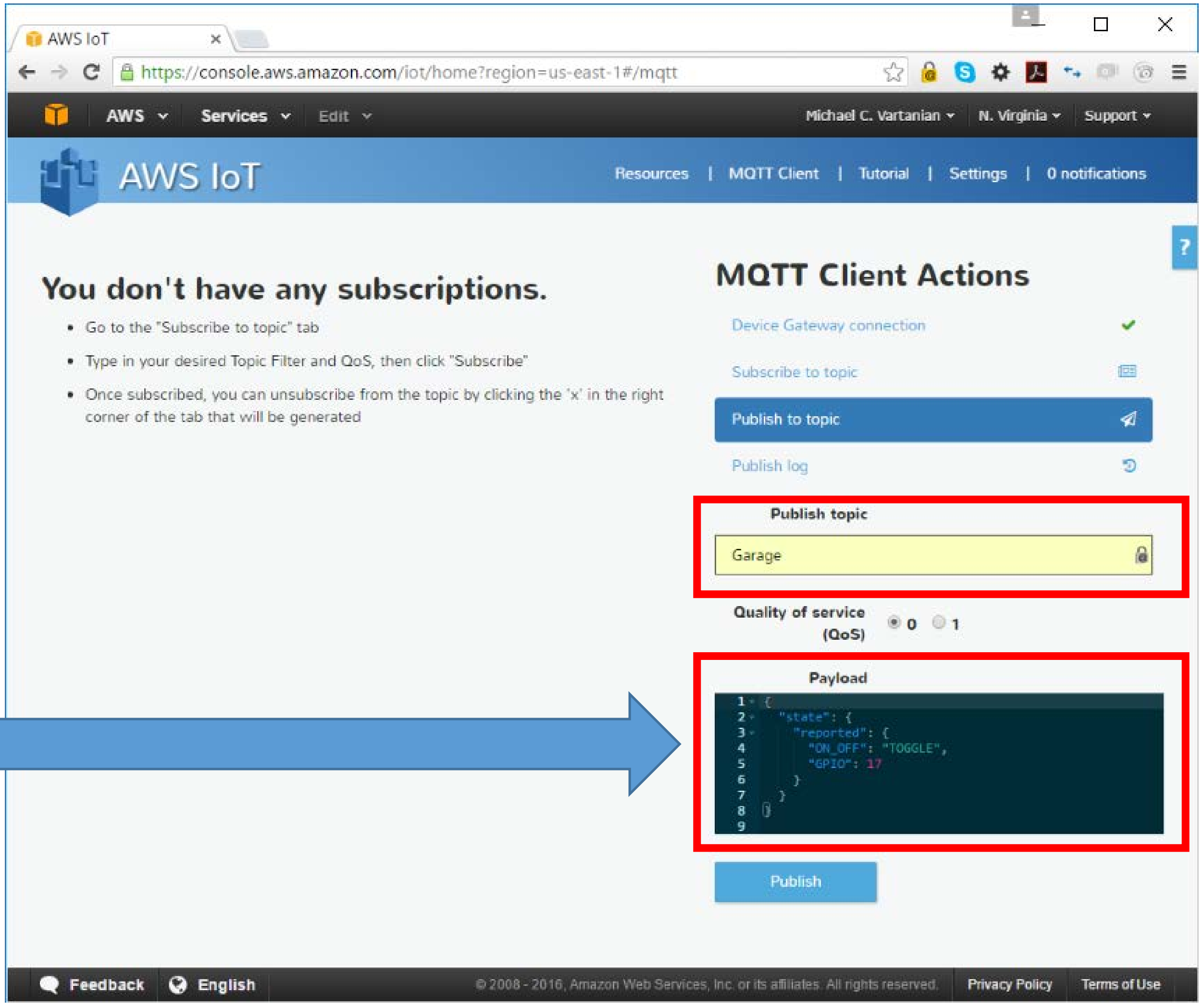
Constants.swift

```swift
import Foundation
import AWSCore

//WARNING: To run this sample correctly, you must set the following constants.
let AwsRegion = AWSRegionType.Unknown // e.g. AWSRegionType.USEast1
let CognitoIdentityPoolId = "YourCognitoIdentityPoolId"
let CertificateSigningRequestCommonName = "IoTSampleSwift Application"
let CertificateSigningRequestCountryName = "Your Country"
let CertificateSigningRequestOrganizationName = "Your Organization"
let CertificateSigningRequestOrganizationalUnitName = "Your Organizational Unit"
let PolicyName = "YourPolicyName"
```


AWS
Services
Edit
Michael C. Vartanian
N. Virginia
Support
AWS IoT
Resources | MQTT Client | Tutorial | Settings | 0 notifications
Resources
Close create panel
Connect AWS IoT Button
Filter by resource names or by resource type (below)
All 1/1 things 0/0 thing types 0/0 rules 1/1 CAs 3/3 certificates 1/1 policies
Select all
Actions
First Previous 1 Next Last
GarageDoorOpener
GarageDoorOpener-Policy
Feedback
English

AWS IoT MQTT Console

- Subscribe to topics
- Publish topic / payload


AWS IoT
https://console.aws.amazon.com/iot/home?region=us-east-1#/mqtt
AWS
Services
Edit
Michael C. Vartanian
N. Virginia
Support
AWS IoT
Resources | MQTT Client | Tutorial | Settings | 0 notifications

You don't have any subscriptions.

- Go to the "Subscribe to topic" tab
- Type in your desired Topic Filter and QoS, then click "Subscribe"
- Once subscribed, you can unsubscribe from the topic by clicking the 'x' in the right corner of the tab that will be generated

MQTT Client Actions

Device Gateway connection
Subscribe to topic
Publish to topic
Publish log

Publish topic: Garage

Quality of service (QoS): [✓ 0] 1

Payload

```
{
  "state": {
    "reported": {
      "ON_OFF": "TOGGLE",
      "GPIO": 17
    }
  }
}
```

Publish
Feedback
English
© 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

Payload shown in JSON format but does not need to be

http://www.w3schools.com/json/json_syntax.asp
http://jsonlint.com/
http://docs.aws.amazon.com/iot/latest/developerguide/thing-shadow-document-syntax.html

Next Steps / Help?

- Code just thrown together for a demo…
  - https://github.com/mvartani76/RPi-AWS-IoT-GarageDoorOpener
- Could use help on the following…
  - Improving iOS Mobile Application
  - Security / Improved Authentication
    - Data logs for usage/alerts
  - Auto run python scripts at startup
    - Linux Crontab, systemd (Jessie)
    - Rc.local, .sh files
  - Garage State Detection Algorithms?
    - Thinking about using Hall Effect Sensor / Magnet at bottom of garage door
    - Should I run wires to this sensor or have an additional wireless node?