IoT Security and Privacy

Assignment 4 – AWS IoT

10 points

Instructions:

1. Note: Blue text points to a web link. Ctrl + Click to follow link.
2. This is a team assignment. However, each member of the team has to submit the finished assignment. Those who do not submit will get zero for this assignment.
3. Answers to all questions must be put into **ONE** document. That is, every time, each student can only submit one report document, answering all questions of this assignment.
4. Students must put answers following each question in this assignment. The instructor will not grade a report with only answers in it and the student gets zero for such an assignment. An assignment report must include original questions.
5. Students MUST submit the finished assignment in either Microsoft Word or pdf format to Blackboard. The doc must be submitted as ONE standalone file and cannot be tarred or zipped into a container.
6. Refer to Print screen on how to take a screenshot. Pressing the Alt key in combination with PrtSc will capture the currently selected window.

Questions:

In this assignment, students are required to connect Raspberry Pi to Amazon AWS IoT and update the state of the connected sensor continuously with the AWS IoT thing shadow. The data such as the state generated by the sensor should be written into Amazon DynamoDB. Note: be careful not to exceed the free account quota of data for AWS IoT. Please refer to the references [1][3][4][5][6][7][8][8], particularly [1], if necessary for this assignment.

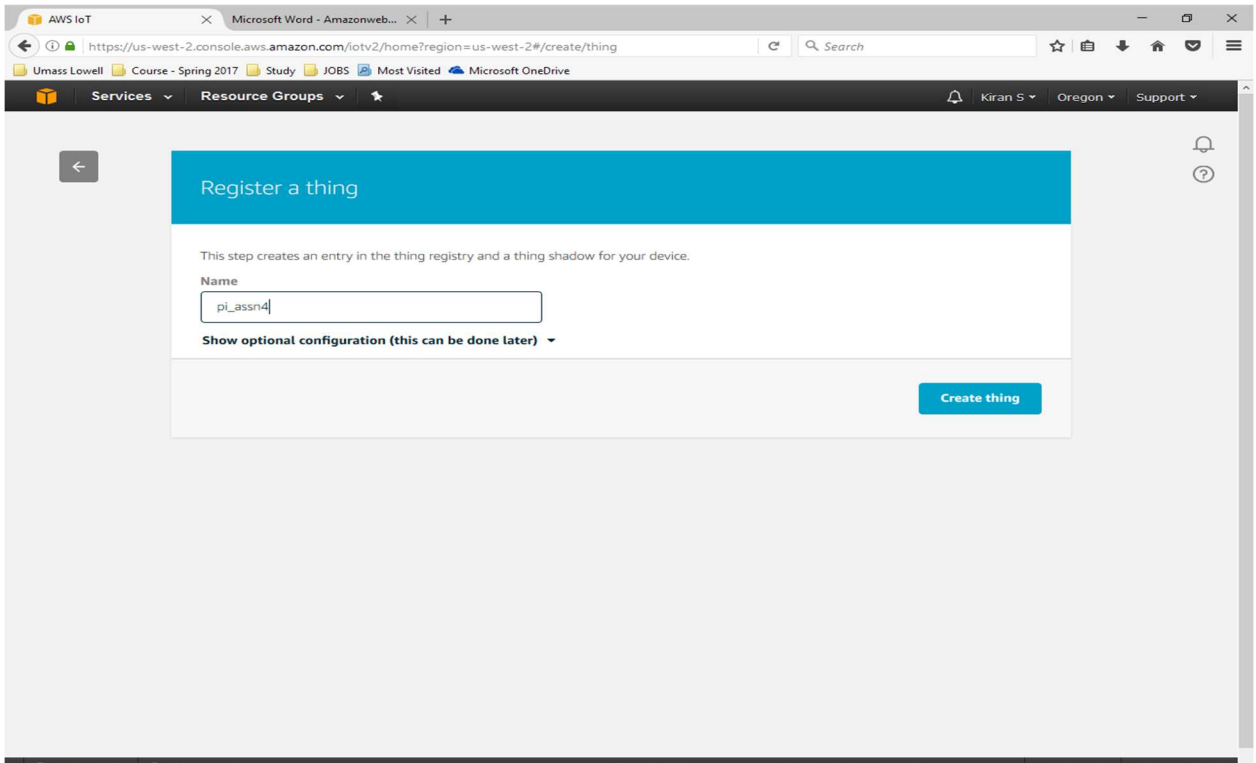
Team members:

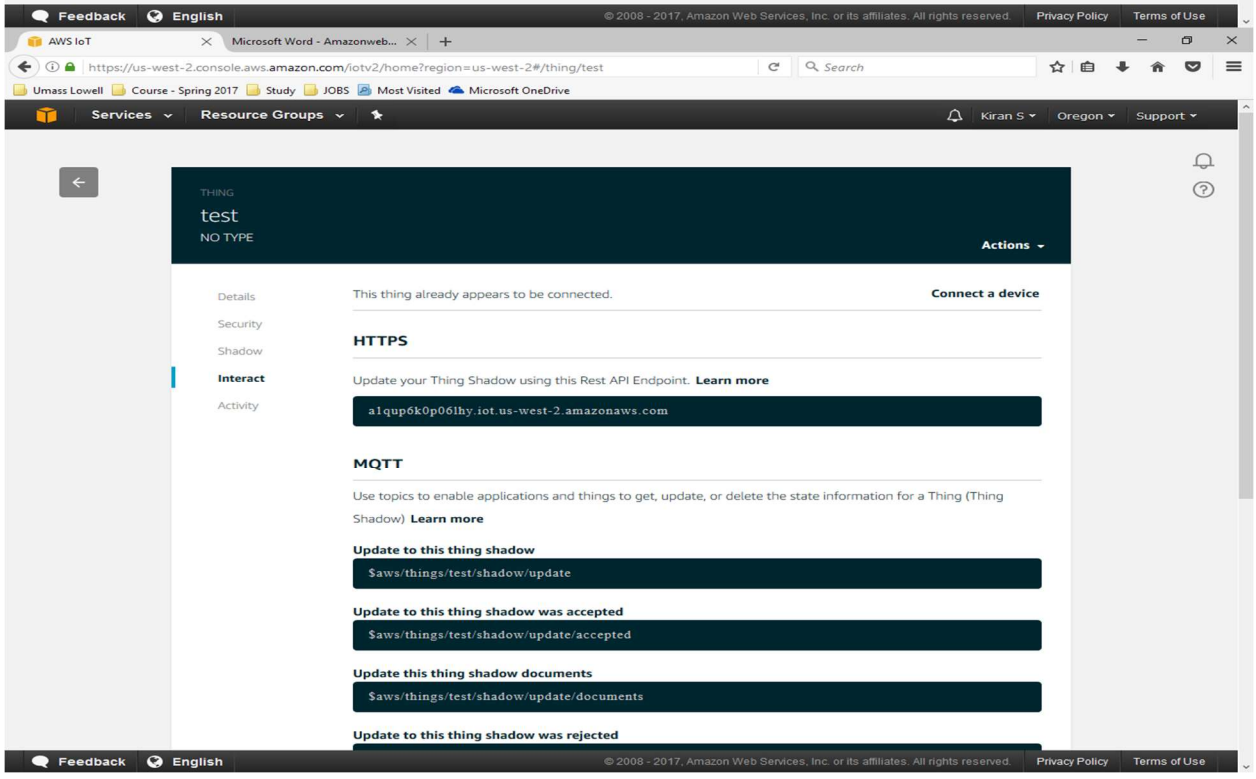
1. **Kiran Shettar**
2. **Tarun Jaykumar Moorjani**
3. **Omkar Salunke**
4. **Rohan Girase**

Requirements:

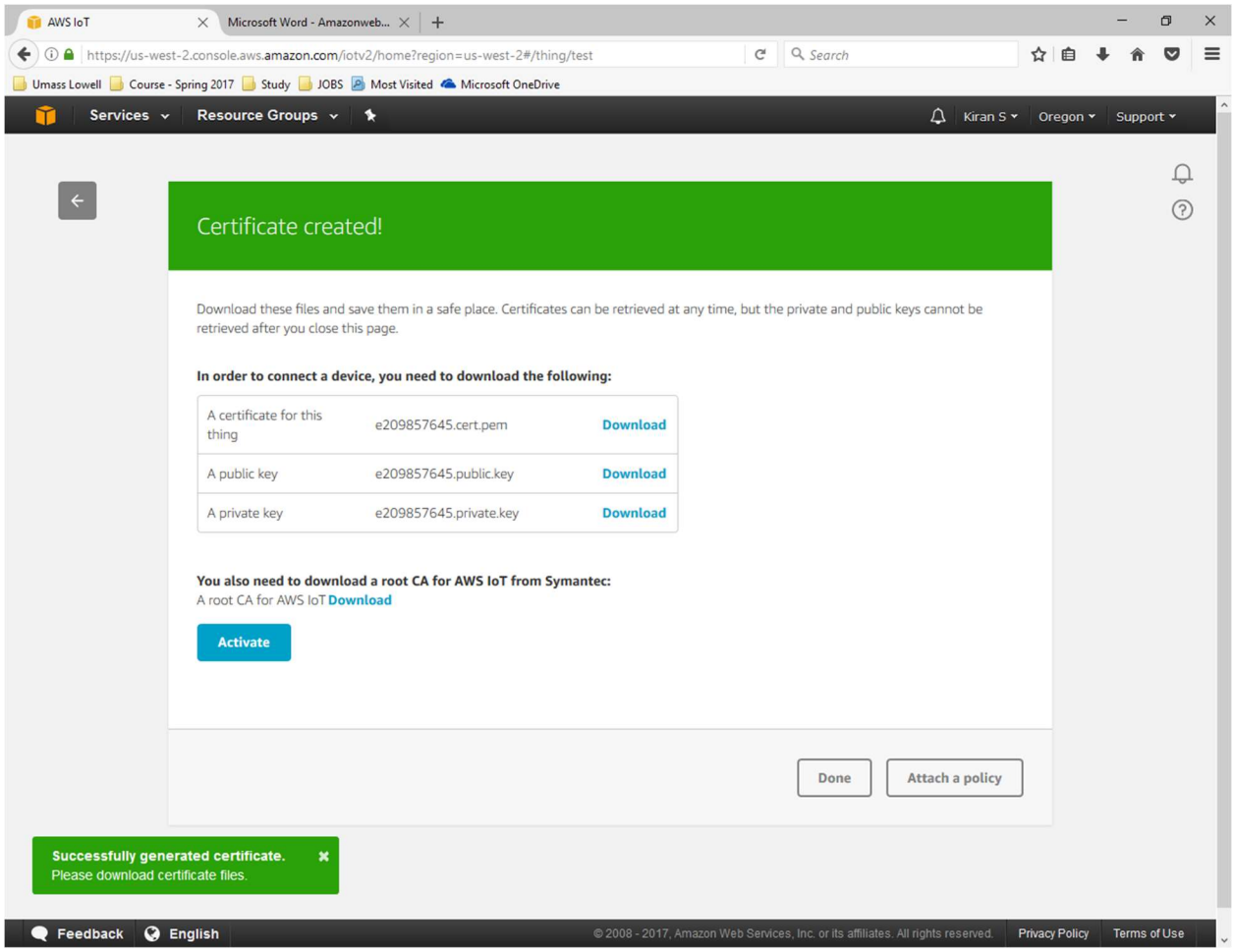
1. Document in detail the procedures connecting Raspberry Pi to Amazon AWS IoT. (3 points)
    - Go to https://aws.amazon.com/iot/ and click on **Sign In to the Console**
    - Create an account and sign in with your account
    - Click on **AWS IoT** under **Internet of Things**

- Go to **Registry** > **Things**. Then give a name for your thing **(pi_assn4)**. Then click on create a thing. After creating a thing Go to **Interact** and copy the **data under HTTPS** and paste it under **mqttsubscriber.py** file to **update our Thing Shadow** using this. It's shown in below image:

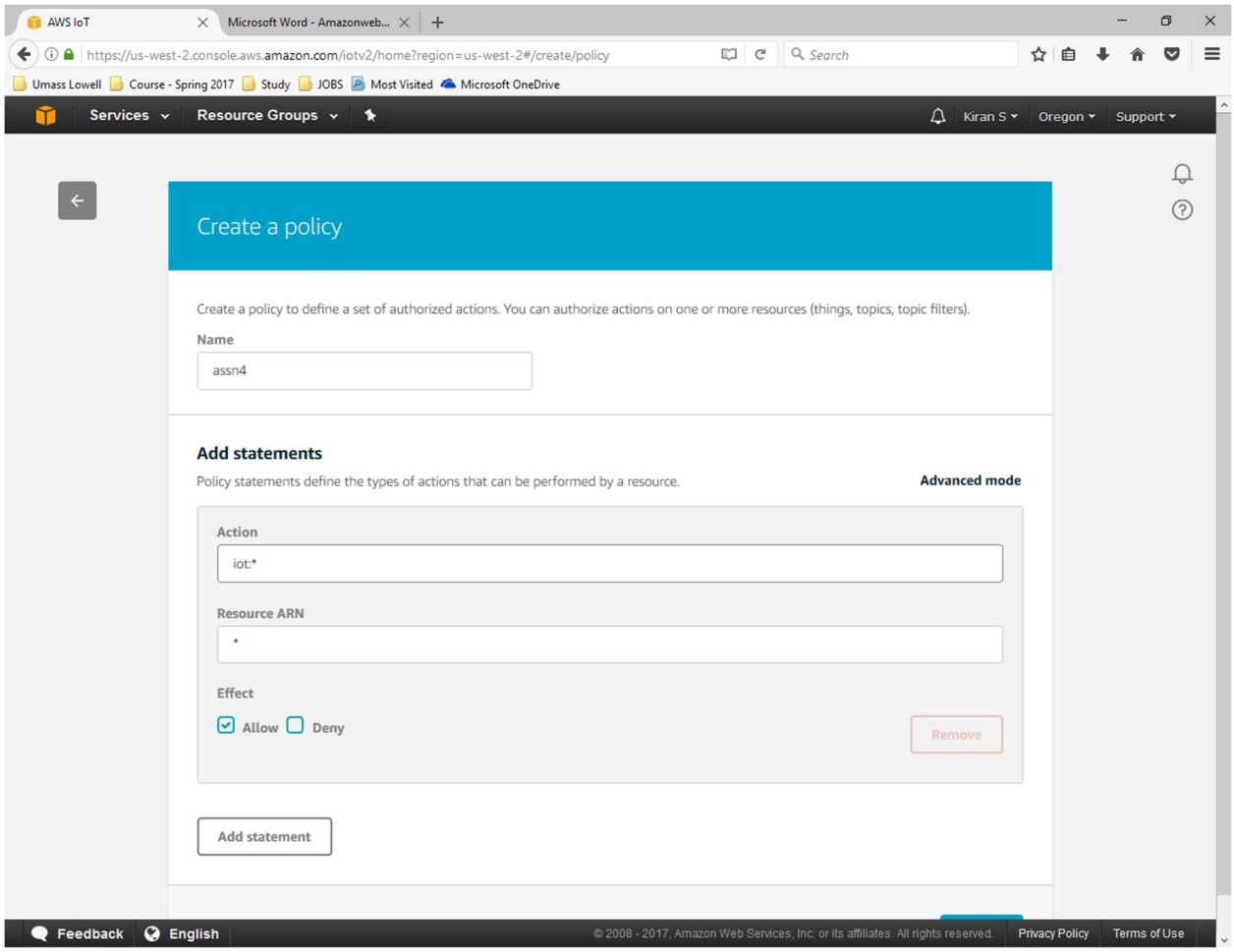




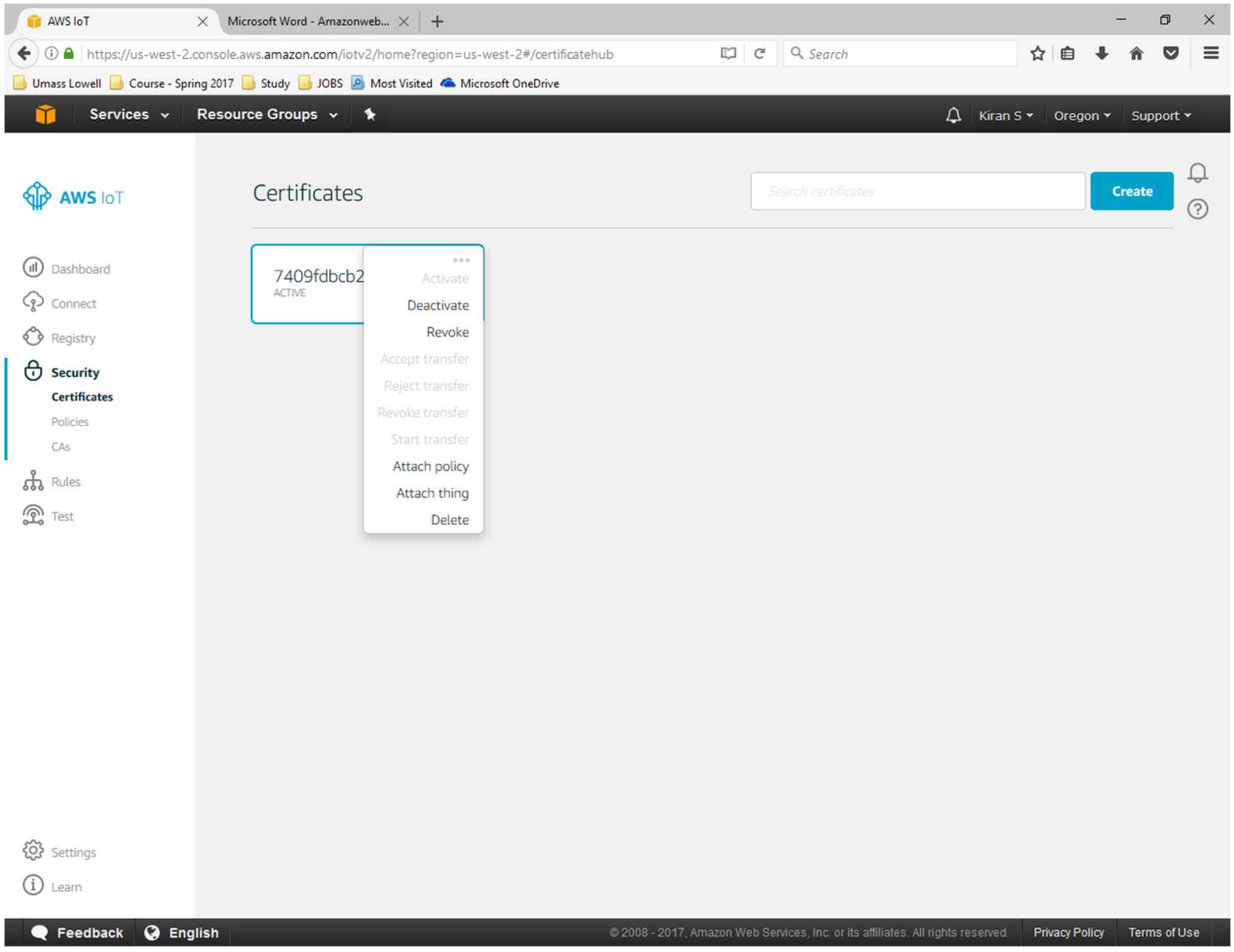
- Go to **Security** > **Create a certificate**. You've to download different files. Those files will be **A certificate for this thing**, **A public key**, **A private key** and **A root CA for AWS IoT (from Symantec)**. Then click on **Activate** as shown in below image:

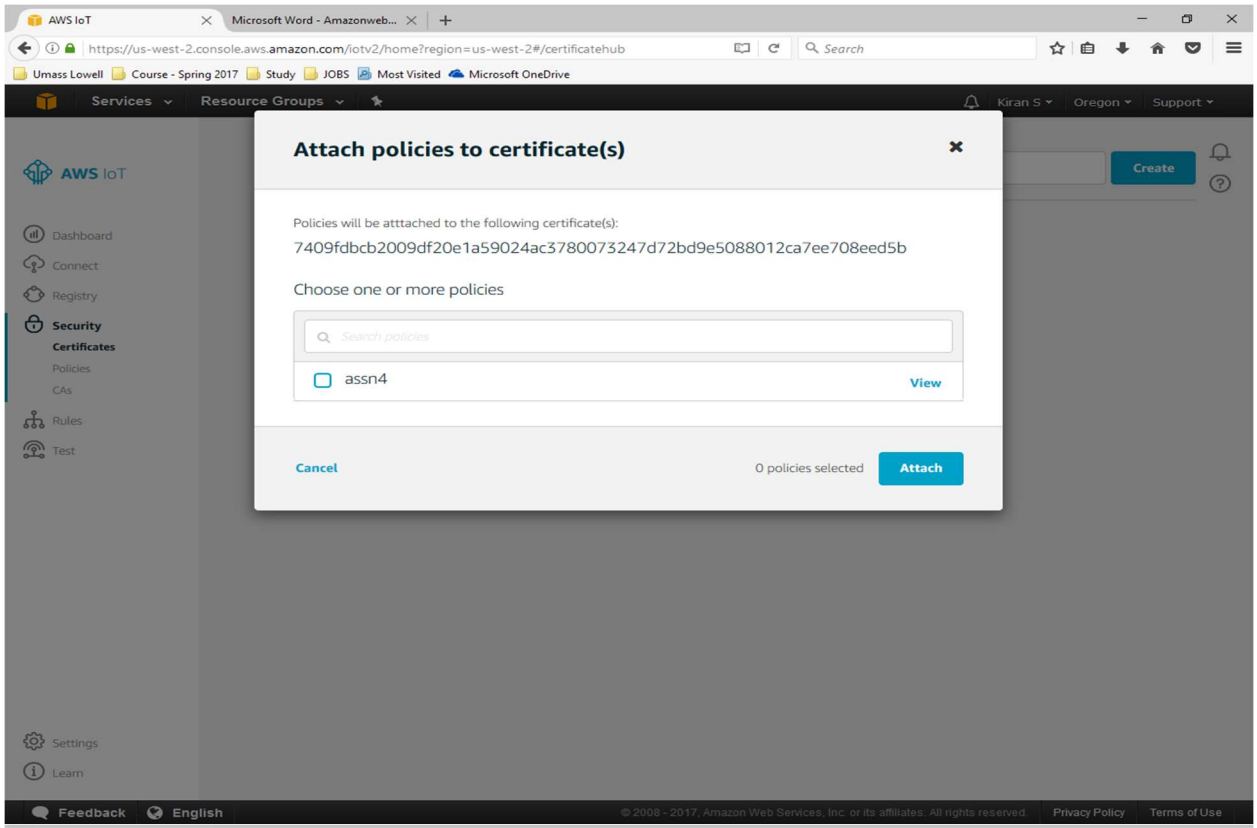


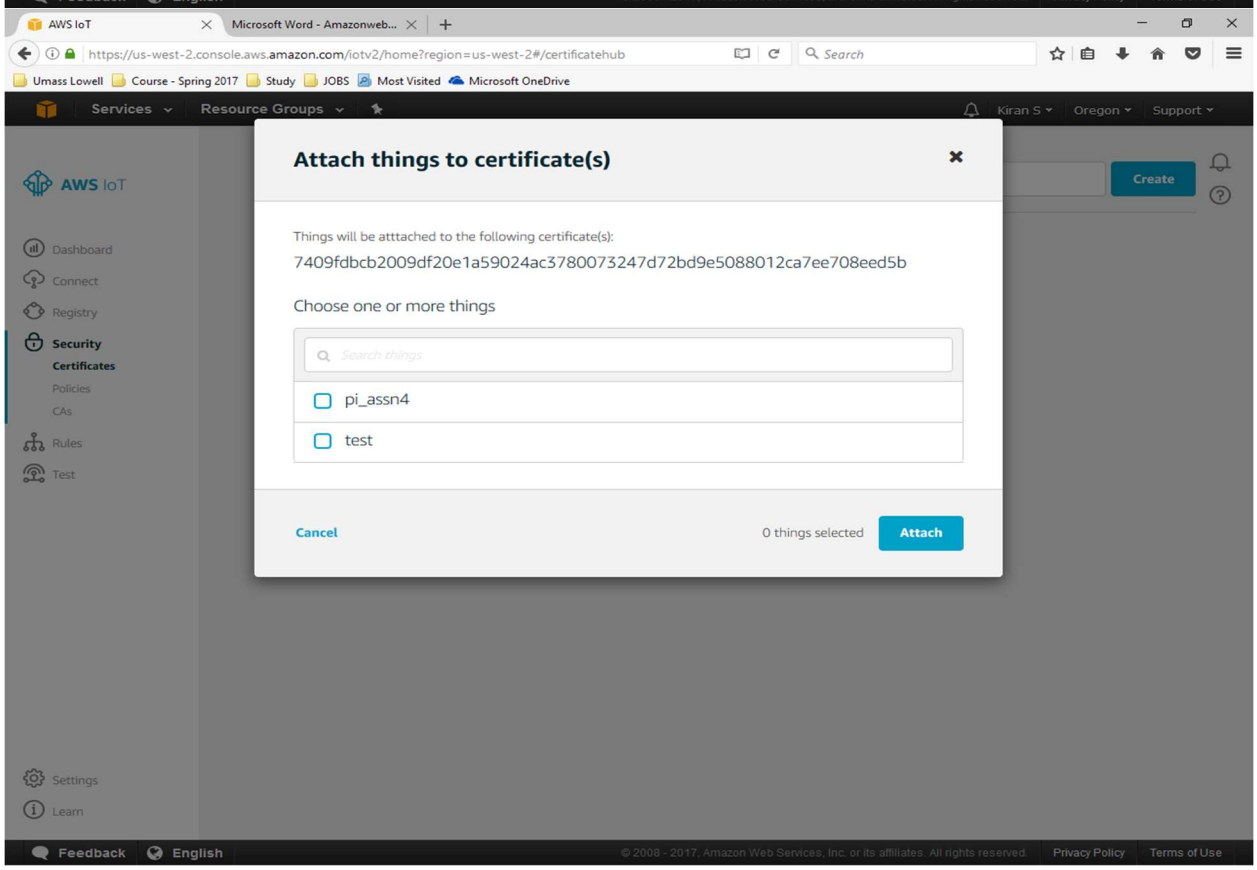
- Go to **AWS IoT** > **Security** > **Policies** then click on **Create.** Give a policy **name, Action: iot*** and **Resource ARN: *** and then click **Create (assn4)** as shown in below image:



- Go to **Security** > **Certificates**. You will find a certificate that's already created. Go to **more options** under the **certificate** created. Then click on **Attach policy**. Then attach the policy that is already created **(assn4)**. Again, in more options under the certificate, click on **Attach thing** then choose the **thing** created **(pi_assn4)** as shown in below image:




AWS IoT
Microsoft Word - Amazonweb...
https://us-west-2.console.aws.amazon.com/iotv2/home?region=us-west-2#/certificatehub
Search
Umass Lowell
Course - Spring 2017
Study
JOBS
Most Visited
Microsoft OneDrive
Services
Resource Groups
Kiran S
Oregon
Support
AWS IoT
Create
Dashboard
Connect
Registry
Security
Certificates
Policies
CAs
Rules
Test
Attach policies to certificate(s)
Policies will be atttached to the following certificate(s):
7409fdbcb2009df20e1a59024ac3780073247d72bd9e5088012ca7ee708eed5b
Choose one or more policies
Search policies
assn4
View
Cancel
0 policies selected
Attach
Settings
Learn
Feedback
English
© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use


AWS IoT
Microsoft Word - Amazonweb...
https://us-west-2.console.aws.amazon.com/iotv2/home?region=us-west-2#/certificatehub
Search
Umass Lowell
Course - Spring 2017
Study
JOBS
Most Visited
Microsoft OneDrive
Services
Resource Groups
Kiran S
Oregon
Support
AWS IoT
Create
Dashboard
Connect
Registry
Security
Certificates
Policies
CAs
Rules
Test
Attach things to certificate(s)
Things will be atttached to the following certificate(s):
7409fdbcb2009df20e1a59024ac3780073247d72bd9e5088012ca7ee708eed5b
Choose one or more things
Search things
pi_assn4
test
Cancel
0 things selected
Attach
Settings
Learn
Feedback
English
© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

- Once we are done with all the above things i.e AWS IoT setup, we should connect the Raspberry Pi to AWS IoT. To begin with this, we should install required python library (MQTT Client and Server). We should download a script "iot-mqtt-subscriber.py" for AWS IoT MQTT Subscriber from http://www.cs.uml.edu/~cgao/Pi/iot-mqtt-subscriber.py

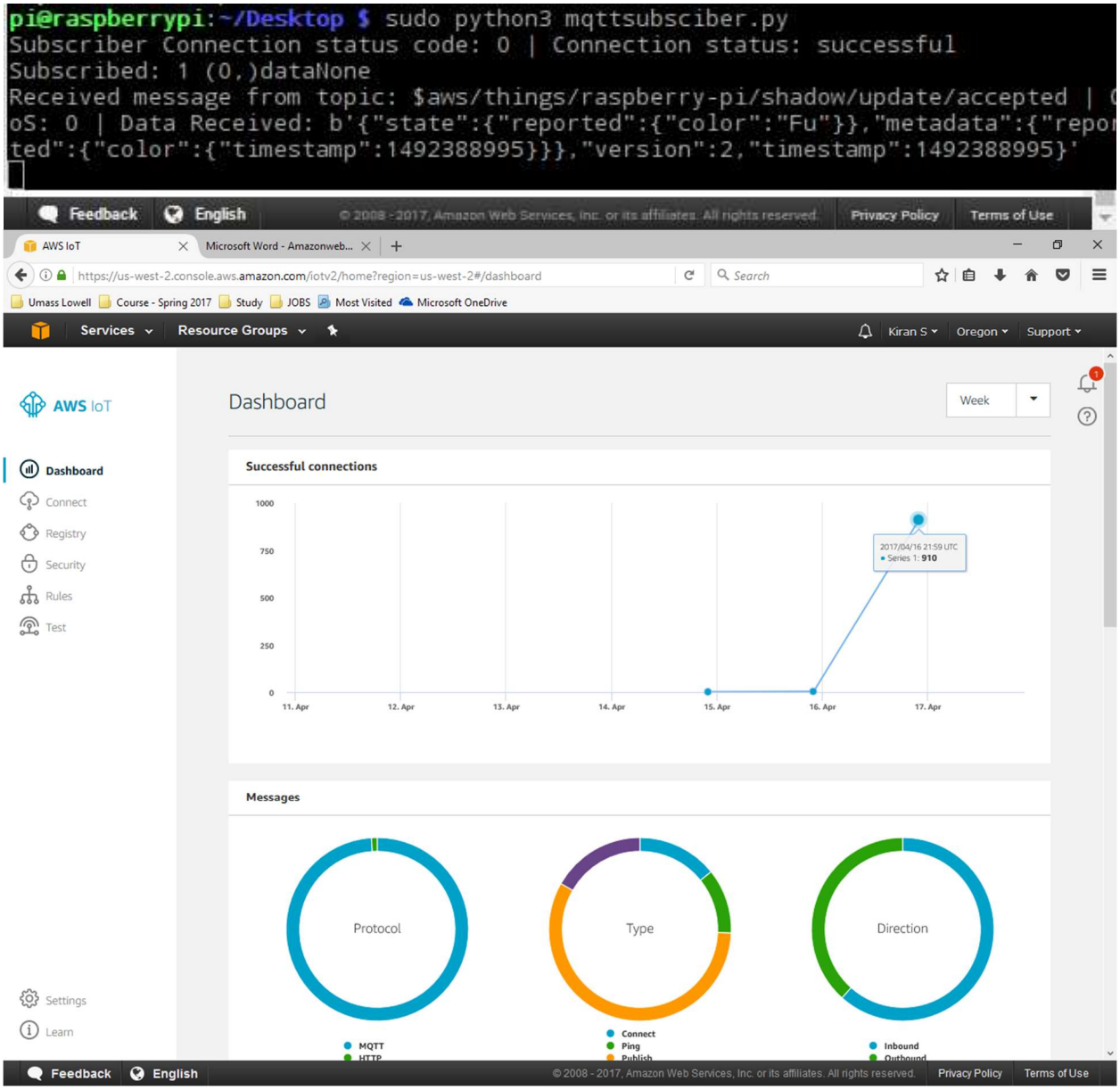
- The subscriber will listen to an AWS IoT topic for messages. We have to make a couple of changes in the code like: **thing name**, **certificate path** and the data we had downloaded **under HTTPS** before.

- The **modified script** for the **mqttsubscriber.py** looks as follows:

C:\Users\kcshe\OneDrive\UML_Files\Spring 2017\IoT - Security and Privacy\5030s2017\assn4\raspbian\mqttsubsciber.py Monday, April 17, 2017 06:12 PM

```
#!/usr/bin/python3

#required libraries
import sys
import ssl
import json
import paho.mqtt.client as mqtt

#called while client tries to establish connection with the server
def on_connect(mqttc, obj, flags, rc):
    if rc==0:
        print ("Subscriber Connection status code: "+str(rc)+" | Connection status: successful")
        mqttc.subscribe("$aws/things/pi_assn4/shadow/update/accepted", qos=0)

        mqttc.publish("$aws/things/pi assn4/shadow/update",'{"state":{"reported":{"color":"Fu"}}}
        ')
    elif rc==1:
        print ("Subscriber Connection status code: "+str(rc)+" | Connection status: Connection
        refused")

#called when a topic is successfully subscribed to
def on_subscribe(mqttc, obj, mid, granted_qos):
    print("Subscribed: "+str(mid)+" "+str(granted_qos)+"data"+str(obj))

#called when a message is received by a topic
def on_message(mqttc, obj, msg):
    print("Received message from topic: "+msg.topic+" | QoS: "+str(msg.qos)+" | Data Received:
    "+str(msg.payload))

#creating a client with client-id=mqtt-test
mqttc = mqtt.Client(client_id="cgao")

mqttc.on_connect = on_connect
mqttc.on_subscribe = on_subscribe
mqttc.on_message = on_message

#Configure network encryption and authentication options. Enables SSL/TLS support.
#adding client-side certificates and enabling tlsv1.2 support as required by aws-iot service
mqttc.tls_set(ca_certs="/home/pi/Desktop/rootCA.pem.crt",
          certfile="/home/pi/Downloads/7409fdbcb2-certificate.pem.crt",
          keyfile="/home/pi/Downloads/7409fdbcb2-private.pem.key",
              tls_version=ssl.PROTOCOL_TLSv1_2,
              ciphers=None)

#connecting to aws-account-specific-iot-endpoint
mqttc.connect("a1qup6k0p06lhy.iot.us-west-2.amazonaws.com", port=8883) #AWS IoT service
hostname and portno

#automatically handles reconnecting
mqttc.loop_forever()
```

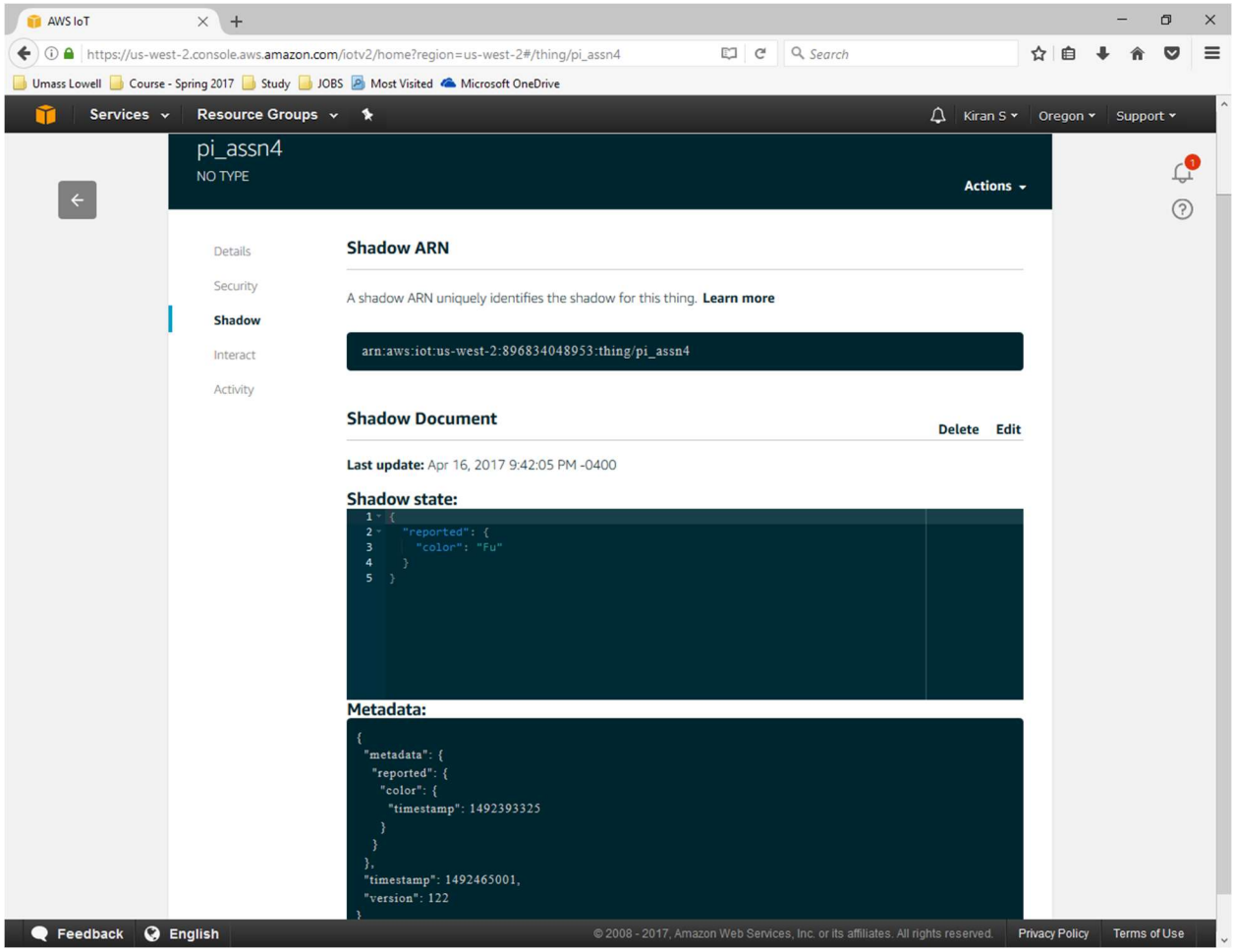
- Then we should run the mqttsubscriber.py file using the command below
  sudo python3 mqttsubscriber.py
  Later you will see the connection status as it is shown in the below image:



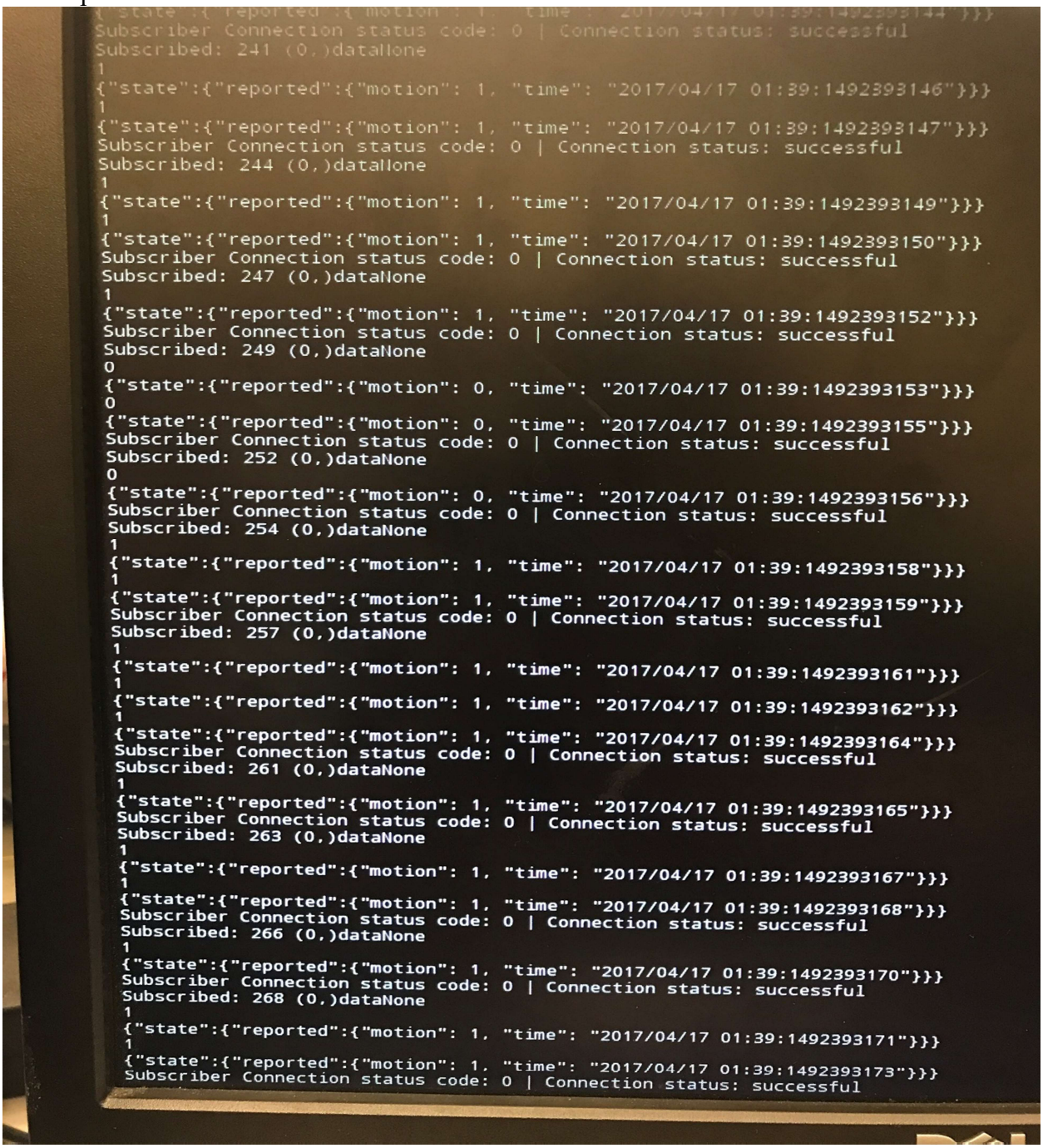
- Hence we make sure that the Raspberry Pi is connected and we update the sensor data with AWS IoT thing shadow. **Continued in next answer**

2. Document in detail the procedures updating the state of the connected sensor continuously with the AWS IoT thing shadow. (3 points)
    - Once we make sure that the Raspberry Pi is connected, we update the sensor data with AWS IoT thing shadow. For that we update the sensor code in **iot-mqttpublisher.py** file. We run this script in a separate terminal as soon as we run the **mqttsubscriber.py** file to detect the motion. We run the below command:
      sudo python3 iot-mqtt-publisher.py

    - The messages received by the subscriber will be shown on the console. As everything worked fine, we can see the status of device on IoT console as shown in below image:

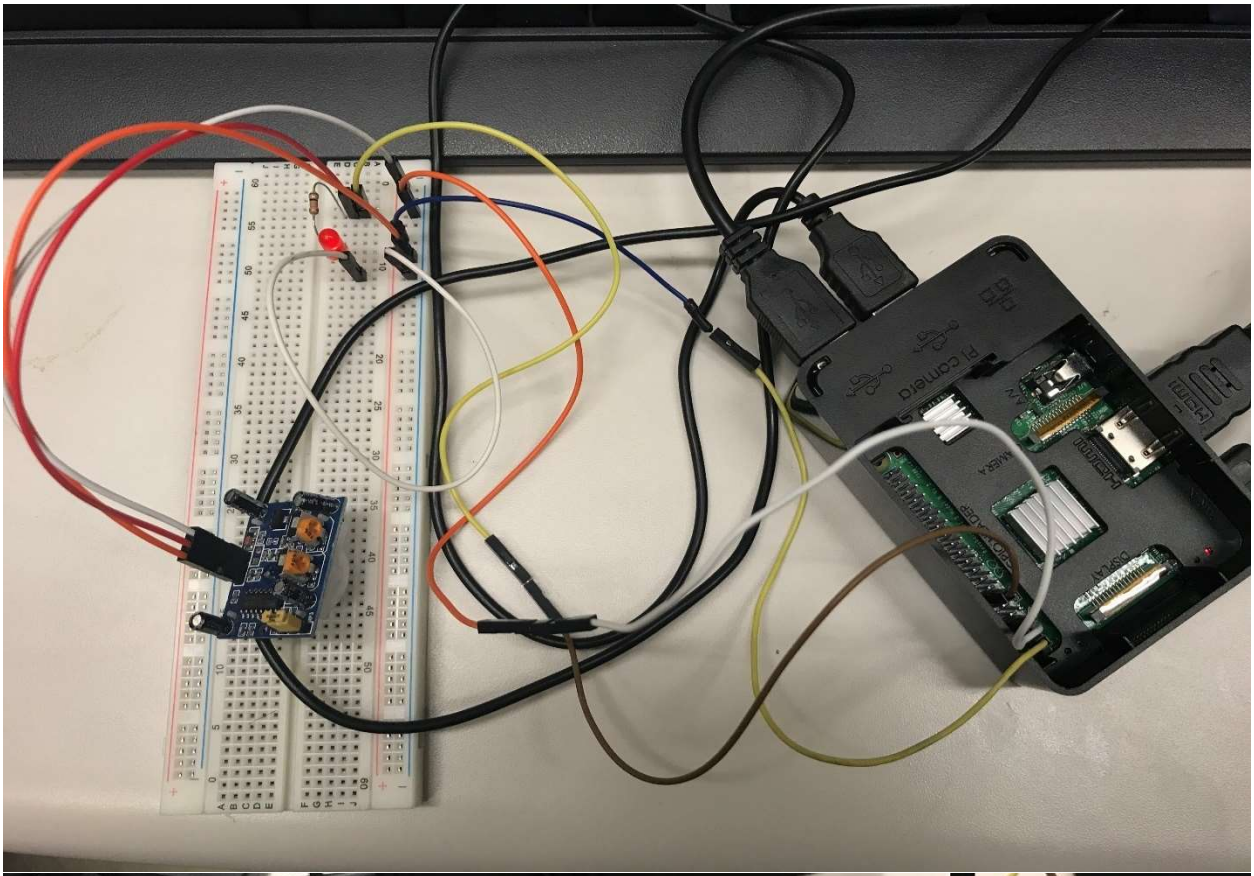


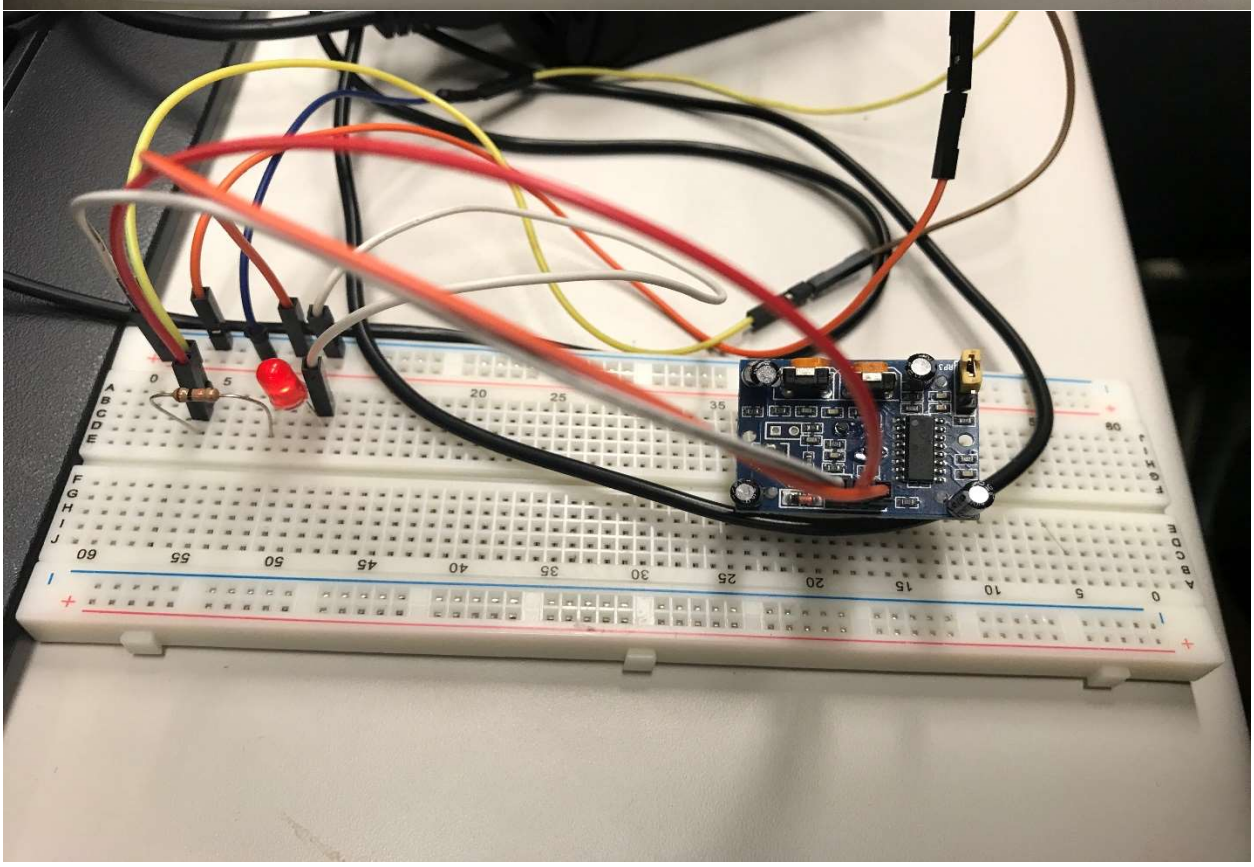
- **PIR sensor motion detection result** looks as follows, when we run the publisher script:



- We are using PIR sensor in our project. Hence we have updated the PIR sensor code. The sensor data is stored on AWS IoT thing shadow.

- Below is the **PIR motion detection sensor connection to the Raspberry Pi:**





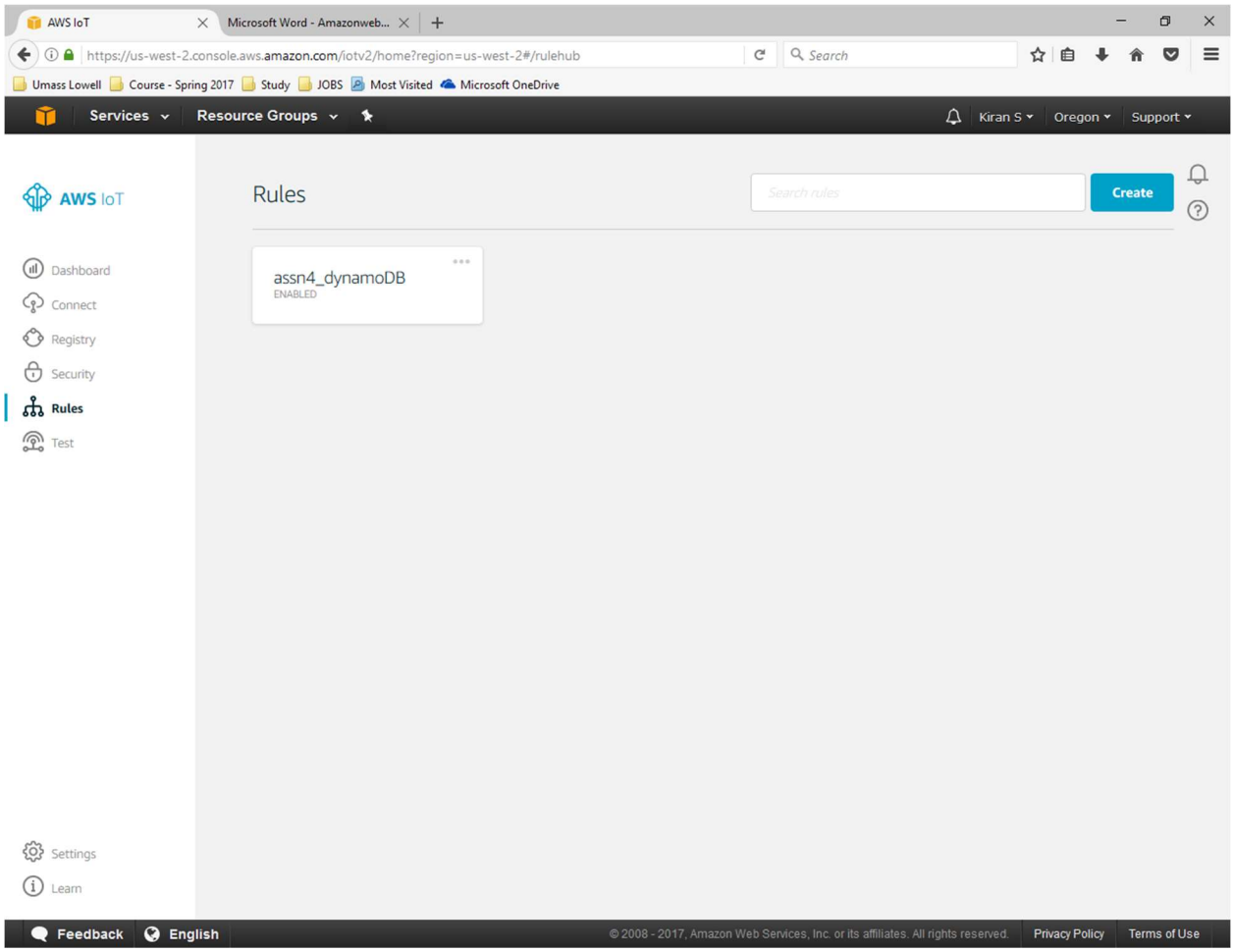
- The **modified script** for the **iot-mqtt-publisher.py** looks as follows:

C:\Users\kcshe\OneDrive\UML_Files\Spring 2017\IoT - Security and Privacy\5030s2017\assn4\raspbian\iot-mqtt-publisher.py Monday, April 17, 2017 06:08 PM

```
#!/usr/bin/python3

#required libraries
import sys
import ssl
import json
import paho.mqtt.client as mqtt

# for motion sensor
import RPi.GPIO as GPIO
import time
from datetime import datetime


#called while client tries to establish connection with the server
def on_connect(mqttc, obj, flags, rc):
    if rc==0:
        print ("Subscriber Connection status code: "+str(rc)+" | Connection status: successful")
        mqttc.subscribe("$aws/things/pi_assn4/shadow/update/accepted", qos=0)
    elif rc==1:
        print ("Subscriber Connection status code: "+str(rc)+" | Connection status: Connection
        refused")

#called when a topic is successfully subscribed to
def on_subscribe(mqttc, obj, mid, granted_qos):
    print("Subscribed: "+str(mid)+" "+str(granted_qos)+"data"+str(obj))

#called when a message is received by a topic
def on_message(mqttc, obj, msg):
    print("Received message from topic: "+msg.topic+" | QoS: "+str(msg.qos)+" | Data Received:
    "+str(msg.payload))

#creating a client with client-id=mqtt-test
mqttc = mqtt.Client(client_id="cgao")

mqttc.on_connect = on_connect
mqttc.on_subscribe = on_subscribe
mqttc.on_message = on_message

#Configure network encryption and authentication options. Enables SSL/TLS support.
#adding client-side certificates and enabling tlsv1.2 support as required by aws-iot service
mqttc.tls_set(ca_certs="/home/pi/Desktop/rootCA.pem.crt",
              certfile="/home/pi/Downloads/7409fdbcb2-certificate.pem.crt",
              keyfile="/home/pi/Downloads/7409fdbcb2-private.pem.key",
             tls_version=ssl.PROTOCOL_TLSv1_2,
             ciphers=None)

#connecting to aws-account-specific-iot-endpoint
mqttc.connect("a1qup6k0p06lhy.iot.us-west-2.amazonaws.com", port=8883) #AWS IoT service
hostname and portno

#automatically handles reconnecting
#start a new thread handling communication with AWS IoT
mqttc.loop_start()

sensor = 12

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(sensor,GPIO.IN)

rc=0
try:
    while rc == 0:
        i = GPIO.input(sensor)
```

3. Document in detail the procedures writing sensor data into Amazon DynamoDB. (3 points)
    - To save sensor's data into AWS database, we should create a rule first. So, we go to **Rules** and click on **Create**. Give some name **(assn4_dynamoDB)** and provide a small Description **(PIR Sensor)** for creating a rule. In **Attribute** area type **#** and in Topic filter area type *****. Then click on **Add Action** > **Insert a message into DynamoDB table** > **Configure action** > **Create rule** > **Choose a role** > **Add action** > **Choose a resource** > **Create a new resource**.

    - The below image shows that the **rule and resource is created**:

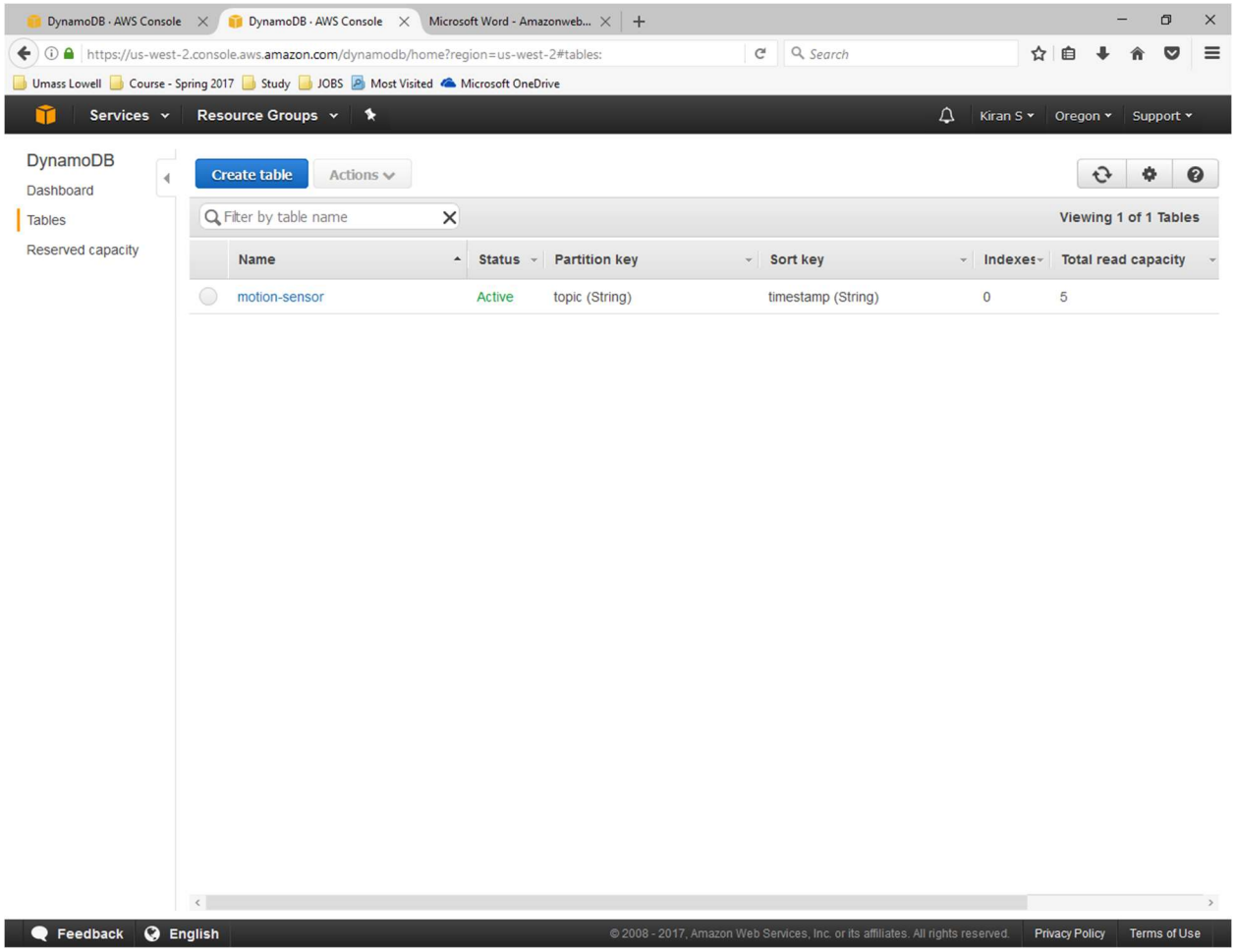


    - Click on **Create table** > **Table name (motion-sensor)** > **Primary Key** fill **topic** and **timestamp.**

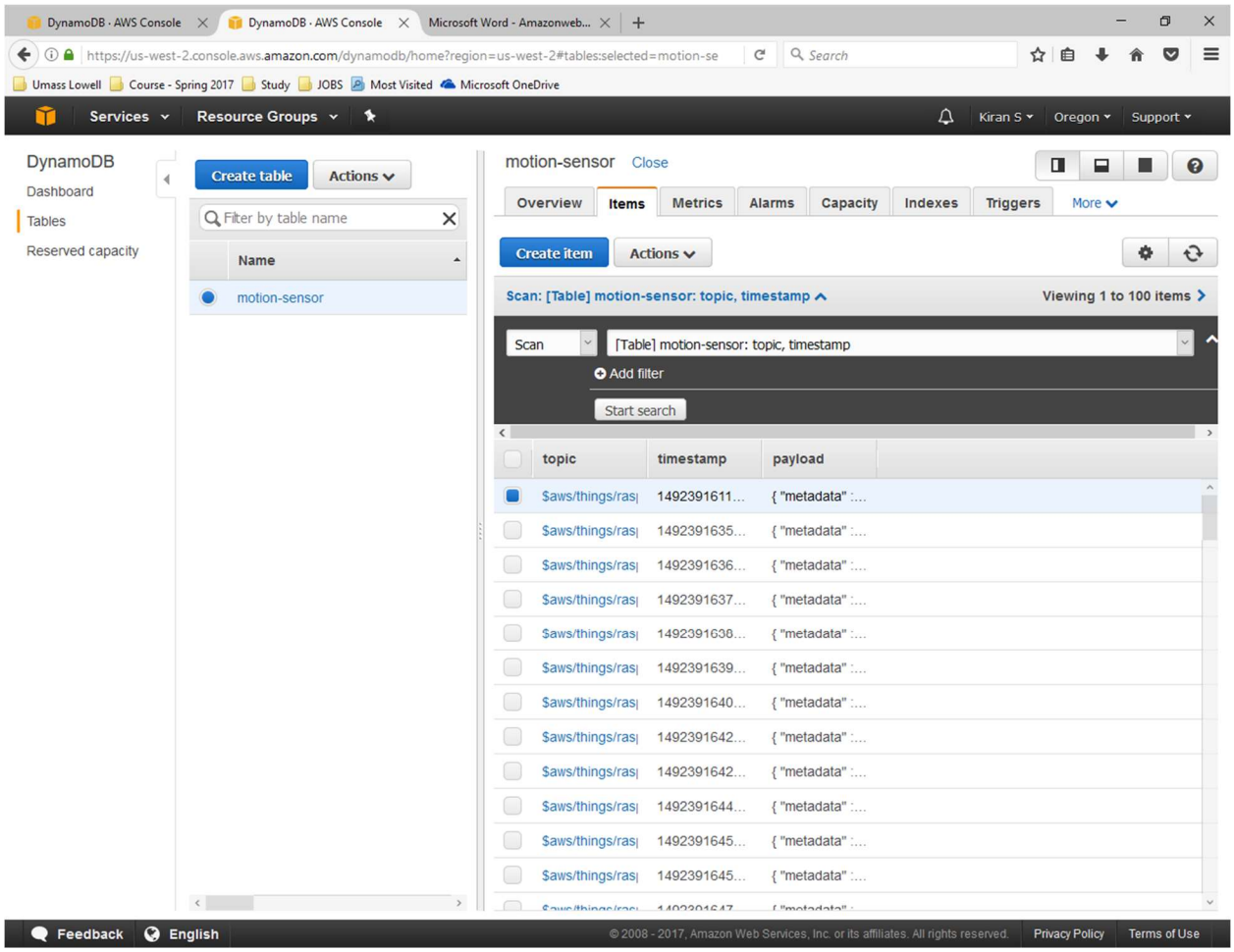
    - Insert **Hash key value** as **${topic()}** and **Range key value** as **${timestamp()}**

    - Go to **Create a new role**, then in IAM management console, choose **Create a new IAM Role** > **Role name** as **aws_iot_motion_sensor_data**. Then click on Allow.

- In the below image, we can see that the table is created:



- Click on **motion-sensor** > **Items**. Here we can see the captured metadata and the data being written on the AWS IoT DynamoDB table as pictured below:



- Hence we can see writing sensor data into Amazon DynamoDB.

4. Post the code for this project into this report. Students should try to write the code for the proposed project in this assignment. (1 point)

References:

[1] Chao Gao, Amazon AWS IoT, 2016
[2] AWS IoT developer guide, 2016
[3] Onur ŞALK, Amazon Web Services IoT, November 02, 2015
[4] Get Started with AWS IoT and Raspberry Pi, Oct. 18, 2015
[5] AWS January 2016 Webinar Series - Getting Started with AWS IoT, Youtube, Jan 26, 2016
[6] AWS Identity and Access Management User Guide, 2016
[7] paho-mqtt 1.1, 2016
[8] Introducing JSON, 2016