Lab 8: Working with Multiple Internet-of-Things

Instructor: Young H. Cho

In the last lab, you have learned to configure and use a network of Internet-of-Things with a Cloud-based IoT hub. In this lab, you will continue to expand your understanding of such end-to-end IoT solution.

1. Posting Data to ThingsboardIO directly from Termux

In the previous lab, you used Node-RED to collect and post data to your IoT Hub (ThingsboardIO). Given that you have Linux compatible terminal with scripting capability, Node-RED is not the only way to send the data. If you can use smaller programs and scripts to do the same thing that you did with Node-RED, it can save time and computing resource.

Since you don't need Node-RED, you may want to uninstall Node-RED from your phone.

```
> npm -g remove node-red
> npm -g remove node-red-admin
> rm -R ~/.node-red
> pkg remove nodejs nodejs-lts
```

Now, install a command line tool and library for transferring data with URLs named curl

```
> apt-get dist-upgrade
> pkg install curl
```

Review the following website to learn how to post your telemetry data to thingsboard using CURL

https://thingsboard.io/docs/reference/http-api/

As you have done in the last lab, set up your sensors on Thingsboard IO account and then obtain their Access Tokens as well as the address of the Thingboard IO instance on the Cloud. Then you should be able to post your data by typing something similar to the following at the command line.

```
> curl -v -X POST --data "{"temperature":42,"humidity":73}"
https://your.cloudinstance.address:8080/api/v1/access_token/telemetry
--header "Content-Type:application/json"
```

Observe that the data that you posted appears in ThingsboardIO as the latest telemetry
Experiment with scripting languages that you know to automate getting the data from one of the sensors on the Android phone then posting the data to ThingsboardIO.

Include the script in the report.

2. Using Python to Read and Post Sensor Data

Install Python and necessary packages into Termux by typing the following:

```
> apt install python
> python3 -m pip install pimux
> python3 -m pip install requests
> python3 -m pip install json
> python3 -m pip install time
```

With these packages installed, you should be able to write a script to automate reading data from sensors using termux-sensor command. You may want to using a simple text editor like nano to do this. If it is not already installed, type the following:

```
> apt install nano
> nano sensor.py
```

Here are the NANO commands https://www.nano-editor.org/dist/latest/cheatsheet.html
As for an example script.

```
---- sensor.py ----

from pimux import scrip

cmd = scrip.compute(f"termux-sensor -s 'sensor_name' -n 1")
print(cmd)

---- sensor.py ----
```

Running the above program should give you the same result as when you typed in the termux-sensor command.

Now, use Python packages to add timestamp then post the data in JSON format to your Thingsboard IO instance.

```
---- sensor.py ----

import requests, json, time
from pimux import scrip

headers = {"Content-Type": "application/json" }
url=f"http://your.cloudinstance.address:8080/api/v1/access_token/
telemetry"

while (1):
   ts = time.time()
   gpscmd = scrip.compute(f"termux-location")
   senread = scrip.compute(f"termux-sensor -s 'sensor_name' -n 1")
   gps=json.loads(gpscmd["output"])
   sen=json.loads(senread["output"])
   tmp=sen["sensor_name"]
   data=json.dumps({"ts":ts,"latitude":gps["latitude"],"longitude":gp
s["longitude"],"sensor_name":tmp["values"][0]})
   resp=requests.post(url,headers=headers, data=data)

---- sensor.py ----
```

Above script should continuously collect and then send data to Thingboard IO.

Verify that the above script works and then submit the resulting script with the report.

3. Monitoring Pressure using Multiple IoTs in Real-time on Thingsboard IO Dash board

Run scripts like the above, configured specifically for each Android phones you have to collect **Pressure, GPS Coordinates, and Timestamp** data. Configure the Dashboard on the Thingsboard IO to display data from all phones you have in real-time. Include the screenshots of the dashboard in the report.

4. Displaying the Data Using Map Widget

Play with the Map widget to display the data collected by each IoT at specific longitude and latitude on the map. As long as you are able to display the data in terms of location on a map, it really doesn't matter what and how you display.

Include the screenshots of the dashboard in the report.

5. Retrieving data from Thingsboard IO Database

We can query and get your sensor data that is stored in the DATABASE associated with Thingsboard IO. Use the following script to get data from Thingsboard IO. Parse the "data" to extract and reformat the data for your needs. It is important for you to figure out what is going on with the script so that you can modify and use it for your purpose.

```
--- get_sensor.py ---
import requests
import json
import datetime

# duration in hours
dur = 24

# sensor device and labels
keys = "temperature,pressure" # your sensor keys - NEED TO MATCH YOUR CONFIGURATION
dev = "26b780b0-2b12-11ee-b632-1d3c39b59be8" # device - NEED TO MATCH YOUR CONFIGURATION

# begin time need to be adjusted for time zone: PST add 7 hours
beg_time = datetime.datetime(2024, 9, 20, 0, 0, 0)+datetime.timedelta(hours=7)
end_time = beg_time+datetime.timedelta(hours=24)  # 24 hours of collected data

inter = int((end_time-beg_time).total_seconds()*1000) # total interval in miliseconds
btime = beg_time.strftime('%s')+'000' # begin time
etime = end_time.strftime('%s')+'000' # end time

# Get JWT Security Token from Thingsboard IO
jwt = requests.post('http://YOUR_THINGSBOARD_ADDRESS:8080/api/auth/login',
      headers={'Content-Type': 'application/json', 'Accept': 'application/json'},
      json={'username': 'tenant@thingsboard.org', 'password': 'password'}).json()
      ['token']

url = 'http://YOUR_THINGSBOARD_ADDRESS:8080/api/plugins/telemetry/DEVICE/'+dev+'/values/
      timeseries?keys='+keys+'&startTs='+btime+'&endTs='+etime+'&interval='+str(inter)
      +'&limit=1000000000&agg=NONE'

data = requests.get(url, headers={'Content-Type':'application/json','X-Authorization':
       f'Bearer {jwt}'}).json()

# data should contain all your data collected between the specified times
```

Use your Python script to gather pressure, GPS coordinates, and timestamps from all your phones.

Include your Python source in the report.

6. Displaying data as a Website using Folium

Go through the following Tutorial to learn to use Folium

https://realpython.com/python-folium-web-maps-from-data/

Create a program that will process your phone data to estimate an area pressure map at different times. I have spoken regarding this during the lecture and gave some simple examples on how you can create spatial maps for all time given multiple mobile IoTs collecting data from the area.

Create an HTML map using Folium to display your result. Include your source codes for the algorithm, Folium based map generation, and screenshots in the report.