KENNESAW STATE UNIVERSITY

CPE 4040: Data Collection and Analysis

Lab 2: MQTT PUB/SUB on Raspberry Pi

Learning Objectives:

1. Learn to create MQTT Broker on Raspberry Pi
2. Understand basic PUB/SUB messaging using Mosquitto MQTT package
3. Understand how to use MQTT messaging options (***Topics***) with Mosquitto

About Mosquito MQTT:

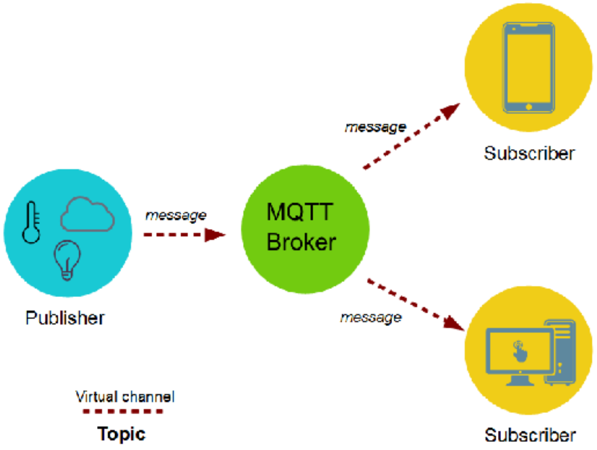
Use links below to learn more about Mosquitto pub/sub commands and available options:

https://mosquitto.org/man/mosquitto_pub-1.html

https://mosquitto.org/man/mosquitto_sub-1.html

Lab Procedure:

1. Power up your Raspberry Pi and open Remote Desktop Connection on your laptop and connect to the Raspberry Pi.
2. After login, open a terminal window. Update and upgrade the Linux packages. It should not take too long this time since you just did it a week ago.
   - Type `sudo apt-get update`
   - Type `sudo apt-get upgrade`
3. We will create a MQTT Client/Broker (Publisher-Subscriber) architecture with **one broker, one publisher and 2 subscribers** on the same Raspberry Pi.



a. Install Mosquitto MQTT broker and client utilities:

```
sudo apt install mosquitto mosquitto-clients
```

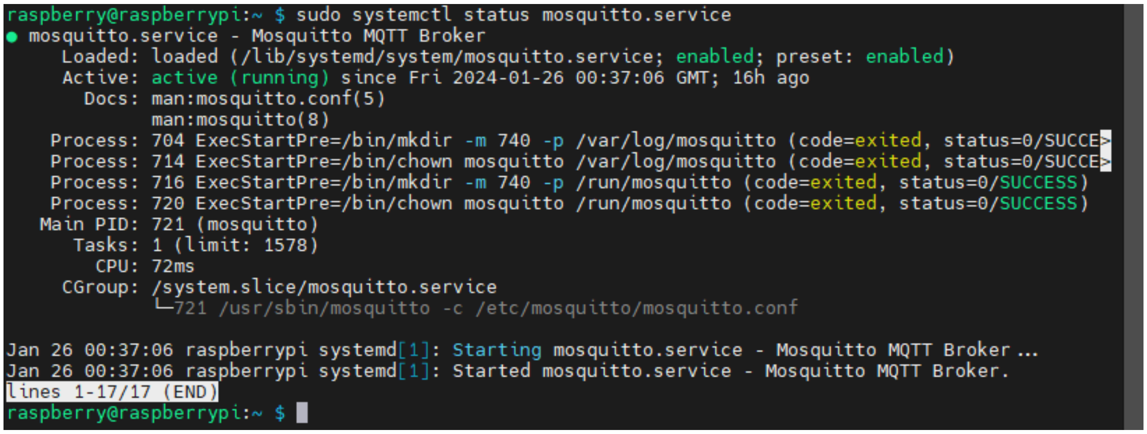
b. Enable the broker and allow it to auto-start after reboot (**Question: What does "systemctl" do?**)

```
sudo systemctl enable mosquitto
```

c. The broker should be running. You can confirm by entering

```
sudo systemctl status mosquitto
```

If it is enabled and running, you will see similar messages like this.



```
raspberry@raspberrypi:~ $ sudo systemctl status mosquitto.service
● mosquitto.service - Mosquitto MQTT Broker
     Loaded: loaded (/lib/systemd/system/mosquitto.service; enabled; preset: enabled)
     Active: active (running) since Fri 2024-01-26 00:37:06 GMT; 16h ago
       Docs: man:mosquitto.conf(5)
             man:mosquitto(8)
    Process: 704 ExecStartPre=/bin/mkdir -m 740 -p /var/log/mosquitto (code=exited, status=0/SUCCE>
    Process: 714 ExecStartPre=/bin/chown mosquitto /var/log/mosquitto (code=exited, status=0/SUCCE>
    Process: 716 ExecStartPre=/bin/mkdir -m 740 -p /run/mosquitto (code=exited, status=0/SUCCESS)
    Process: 720 ExecStartPre=/bin/chown mosquitto /run/mosquitto (code=exited, status=0/SUCCESS)
   Main PID: 721 (mosquitto)
      Tasks: 1 (limit: 1578)
        CPU: 72ms
     CGroup: /system.slice/mosquitto.service
             └─721 /usr/sbin/mosquitto -c /etc/mosquitto/mosquitto.conf

Jan 26 00:37:06 raspberrypi systemd[1]: Starting mosquitto.service - Mosquitto MQTT Broker...
Jan 26 00:37:06 raspberrypi systemd[1]: Started mosquitto.service - Mosquitto MQTT Broker.
lines 1-17/17 (END)
raspberry@raspberrypi:~ $ 
```

d. Before publishing and sending messages to the broker, a few settings need to be configured. In the default directory, enter the following command:

```
sudo nano /etc/mosquitto/mosquitto.conf
```

e. Once in the file, use the arrow keys to navigate to the bottom of the file. Add the following two lines of code at the bottom:

```
listener 1883
allow_anonymous true
```

f. Press Ctrl+X to save and exit the configuration file. Execute the following command:

```
sudo systemctl restart mosquitto
```

4. Open two additional Terminal windows. Now you should have three open terminals. Make a note that one of them is for "Publisher" and the other two are for "Subscriber 1" and "Subscriber 2", respectively.

5. In the Subscriber 1 Terminal, subscribe to a "topic" **/sensors/room1/dev1**

- `mosquitto_sub -h <ip_address> -t /sensors/room1/dev1` (Note: plug in your own Raspberry Pi IP address!)
- After this command, the client will stay in Subscribe mode and wait for messages.

6. In the Subscriber 2 Terminal, subscribe to a “topic” **/sensors/room2/dev1**

   - `mosquitto_sub -h <ip_address> -t /sensors/room2/dev1` (Note: the room number is different!)

7. In the Publisher Terminal, start to publish messages to the newly created topics:

   ```
   mosquitto_pub -h <ip_address> -t /sensors/room1/dev1 -m “10”
   ```

   ```
   mosquitto_pub -h <ip_address> -t /sensors/room2/dev1 -m “20”
   ```

8. Go back to the Terminals for Subscriber 1 and Subscriber 2 to check if the messages are received. In the Publisher Terminal, try to publish to a different topic (e.g., different room or device number) to observe that no message is delivered to the subscribers.

9. **Topic Wildcard**: Go back to Subscriber 1 Terminal and exit the Subscribe mode by pressing Ctrl+C. Then subscribe to another topic: **/sensors/+/dev1**. Note that we are using the **wildcard** “+”. Publish the same two messages as in Step 8. Subscriber1 should receive both messages whereas Subscriber 2 should receive only the second one.

10. **Topic Wildcard**: Exit Subscribe mode for both subscribers then do the following:

    - For Subscriber 1, subscribe to the topic: **/sensors/#.** Note that we are using the **wildcard** “#”.
    - For Subscriber 2, subscribe to the topic: **/sensors/room2/**#.
    - Publish messages similar to those in Step 8, but using different room numbers and device numbers in each case. Observe and explain the results for both subscribers.

11. **Handshaking messages and QoS**: For Publisher and Subscriber 1, enable debug messages using the option “-d” to observe handshaking messages on both sides. Use option “-q” to change the QoS flags (0, 1, 2) on both Publisher and Subscriber and observe the result. Capture the message exchanges and explain the difference between them. For example,

    ```
    mosquitto_sub -h <ip_address> -t /sensors/room1/dev1 -d -q 1
    ```

    ```
    mosquitto_pub -h <ip_address> -t /sensors/room1/dev1 -m “10” -d -q 1
    ```

12. **Retained message**: Go back to Subscriber 2 Terminal and stop the client (if it is still active).

    - In the Publisher Terminal, publish the second message in Step 8 (pertaining to room2). Then go back to Subscriber 2 and subscribe to the same topic. Observe that nothing would be delivered to the subscriber during or after the publishing.
    - Stop the client in Subscriber 2 terminal. The Publisher would publish the same message, however, with an additional “-r” option:

      ```
      mosquitto_pub -h <ip_address> -t /sensors/room2/dev1 -m “30” -r
      ```

    - Start the client in Subscriber 2 and observe that the retained message is delivered immediately when the client is subscribed to that topic.

Post Lab Exercise: Automatic IP Address Email from Raspberry Pi on Boot

Before using Raspberry Pi in SSH or remote desktop mode, you need to figure out the IP address of the WiFi connection, whether at home or on campus. An elegant way to do this is to allow your Raspberry Pi automatically send you an email with the IP address whenever it is powered up.

Please research on how this can be done. You will **briefly describe your approach** and demonstrate the results, including

- A reference link to the original solution that your method is based on.
- A screenshot of the Python script you either created or modified for this task.
- A screenshot of the received email confirming the successful IP address delivery after Raspberry Pi boot-up.

Lab Report:

Be sure to capture screenshots from steps 3 to 12.

Submit the report via D2L Assignment, in PDF format.