IoT Challenge 2 – Exercise

Leader: Ni Giovanni 10831328

Gu Xinyue 10840236

Part 2 - Exercise

A wireless IoT network consists of the following devices:

- A battery-powered, Wi-Fi-enabled **temperature sensor** that measures and transmits temperature data every 5 minutes.
- A battery-powered, Wi-Fi-enabled **valve** that receives temperature readings from the sensor and computes the average temperature every 30 minutes to decide whether to open or close.
- **A Raspberry Pi**, connected to the power grid, which only supports MQTT for communication.

The temperature sensor and valve can communicate using either **MQTT** or **CoAP**, with a specific pre-defined topic or resource. The topic/resource length is 10 bytes and the payload size is 8 bytes. However, since the Raspberry Pi only supports MQTT, any interaction between the Raspberry Pi and the battery-operated devices must use MQTT. The sensor and valve, however, can communicate directly using CoAP if desired.

Part 2 - Exercise

Consider the following message sizes (in bytes), which **already include** header and payload size for the COAP resource or MQTT topic used in the system:

COAP		MQTT	
GET Request	60 B	Subscribe	58 B
GET Response	55 B	Sub Ack	52 B
PUT Request	77 B	Publish	68 B
PUT Response	58 B	Pub Ack	51 B
Empty ACK	14 B	Connect	54 B
		Connect Ack	47 B
		Ping Req	52 B
		Ping Resp	48 B

Part 2 - Exercise

Assuming that:

1. Transmit and Receive cost per bit are
   E_{TX} = 50nJ/bit, E_{RX} = 58nJ/bit (nanojoule per bit)

2. The Wi-Fi network is ideal **(no losses)**

3. The processing cost on the valve to compute the average temperature every 30 minutes is **Ec = 2.4mJ (millijoule)**

4. The sensor and valve start in power-off state

Part 2 - Exercise

Exercise Question 1 (EQ1): Compute the total energy consumed by the two battery-powered devices **over a period of 24 hours** in both cases when using COAP **(a)** and MQTT **(b)**, using each in its **most efficient configuration energy-wise.**

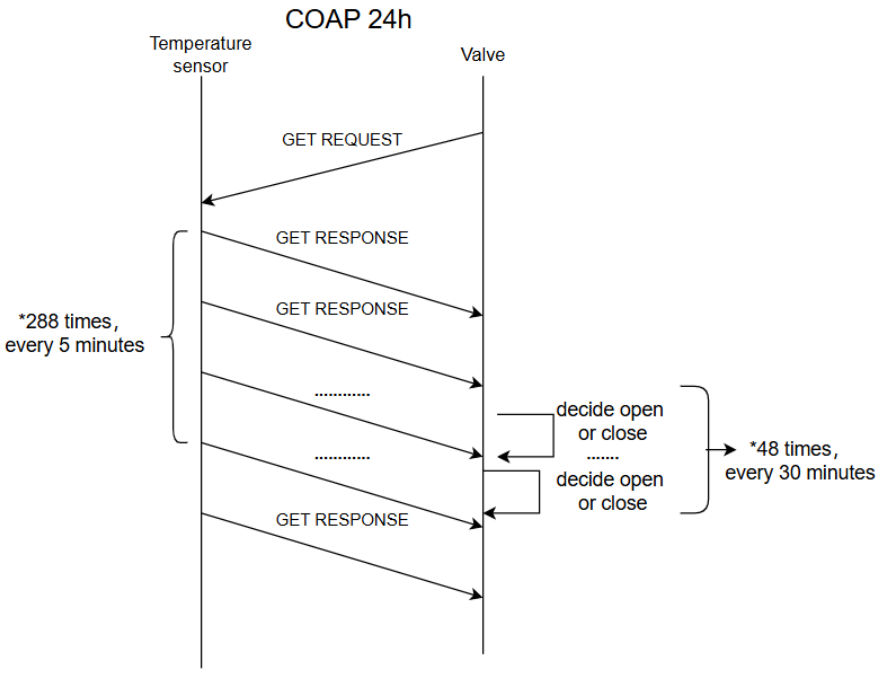
Exercise Question 2 (EQ2): Propose **atleast one** solution for decreasing the energy consumption when passing using the Raspberry PI as a broker. **Give a rough estimate of the energy saving** that could be obtained with your solution: recompute the energy under your proposed configuration.

EQ1

a)

In the first case, since the sensor and the valve can communicate directly using the CoAP protocol, the sensor can act as a CoAP server and the valve as a CoAP client. We believe that using the observe mode is the most efficient configuration, as it eliminates the need for the valve to send a GET request every 5 minutes.

The following figure illustrates our logic. Since we are focusing on energy consumption rather than time, we do not represent transmission delays or timing differences. Our main goal is to highlight the messages exchanged between the two devices.



- The sensor will send temperature information every 5 minutes, so over 24 hours: 24*60/5 = 288 times.

- Therefore, the sensor will transmit 288 times and the valve will receive 288 times.
  Each GET response is 55 Bytes.
  Transmission cost (sensor):

  288*55*8* Etx = 288*(55*8)bit*50nJ/bit = 6.336mJ

  Reception cost (valve):

  288*55*8*Erx = 288*(55*8)bit*58nJ/bit = 7.349mJ

- The valve will send only one GET request, and the sensor will receive it once, Each GET request = 60 Byte
  Transmission cost (valve):

  60*8*Etx = (60*8)bit*50nJ/bit = 0.024mJ

  Reception cost (sensor):

  60*8*Erx = (60*8)bit*58nJ/bit = 0.028mJ

- Processing cost on the valve to compute the average temperature every 30 minutes: Ec = 2.4mJ.

- Since it runs every 30 minutes, over 24 hours: 48 times.
  Total energy:

  48*2.4mJ = 115.2mJ

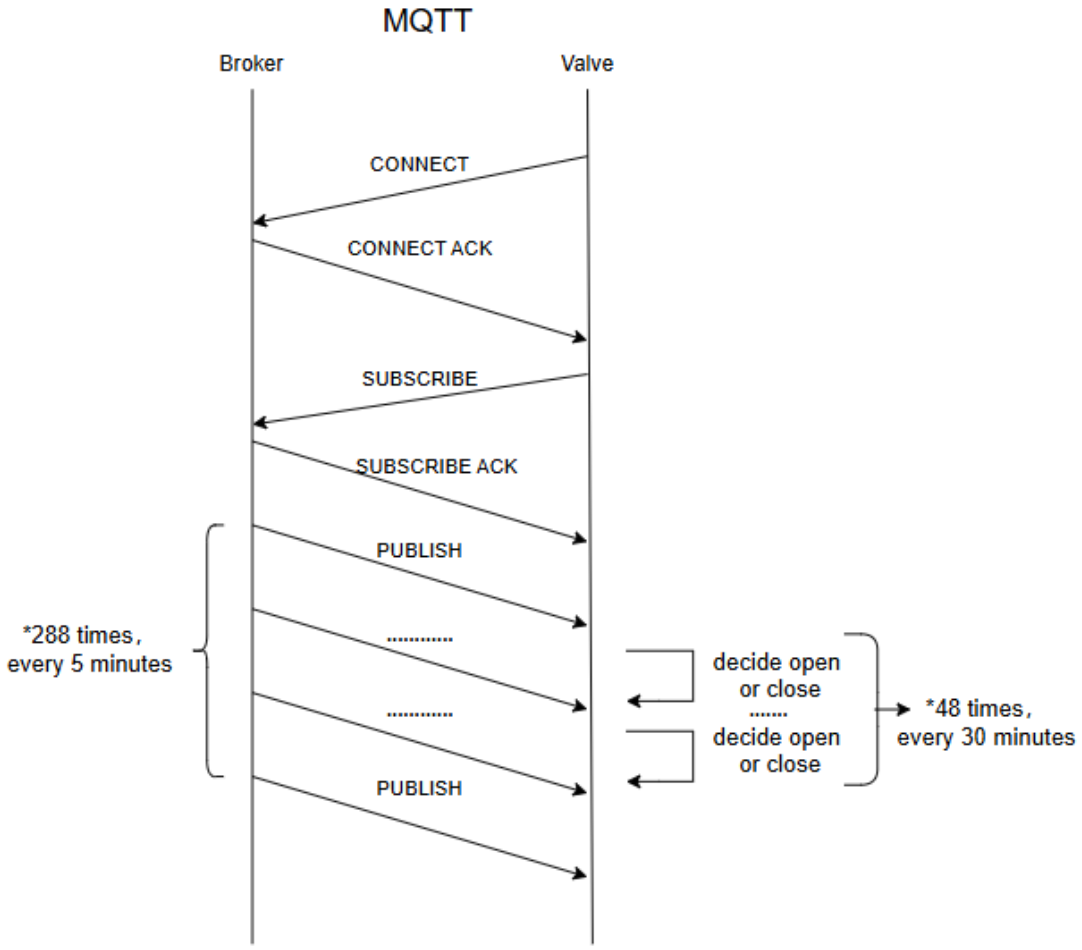
  Total energy consumption:

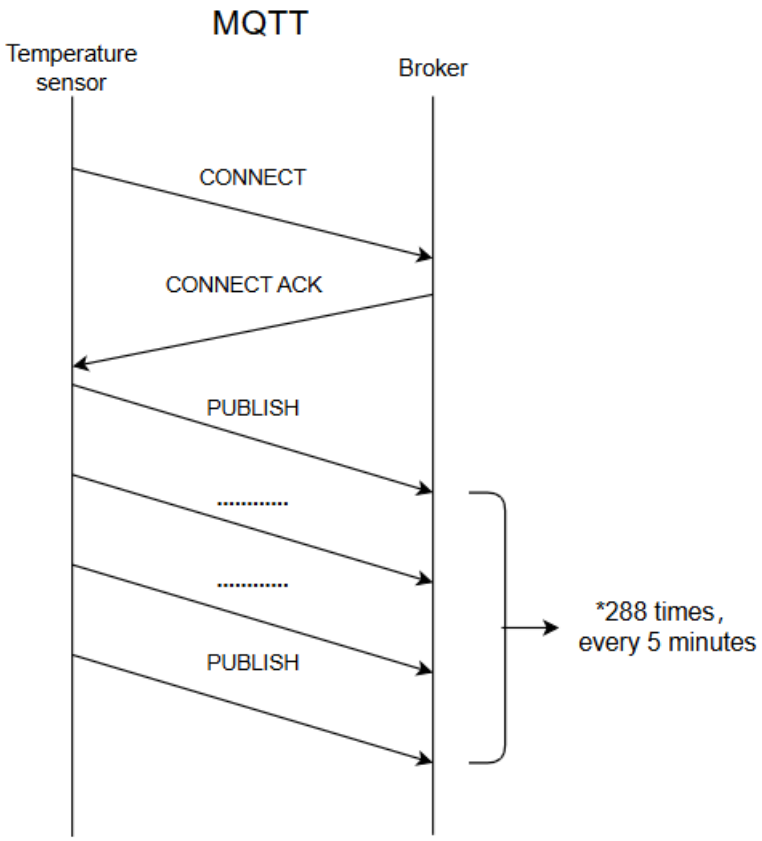
  6.336 + 7.349+ 0.024+ 0.028+ 115.2= 128.937mJ

b)

In the MQTT case, a broker is required between the two sides. We can use the Raspberry Pi provided by the exercise for this purpose. The sensor can act as the publisher, and the valve as the subscriber to that resource. We believe that QoS level 0 is the most energy-efficient configuration. In this mode, the publisher does not receive a PUBLISH acknowledgment (ACK). Since the sensor publishes 288 times a day, avoiding these ACKs can save a significant amount of energy. However, this comes at the cost of reliability, as QoS 0 does not guarantee that the subscriber will successfully receive all messages.

The following figures represent our logic. As we are focusing on energy consumption rather than time, transmission delays and time intervals are not shown, we mainly highlight the messages exchanged.





As in case 1a, the sensor publishes 288 times over 24 hours, and the valve computes the average temperature 48 times during the same period.

Both the sensor and the valve send a CONNECT message and receive a CONNACK message.
Transmission cost:

$$2 * Connect\_Size * Etx = 2 * (54 * 8)bit * 50nJ/bit = 0.0432mJ$$

Reception cost:

$$2* Connect\_ACK\_Size * Erx = 2*(47*8)*58nJ/bit = 0.0436mJ$$

- The sensor publishes the information 288 times:
  Transmission cost:

$$288*Pub\_Size * Etx = 288*(68*8)\ bit*50nJ/bit = 7.8336mJ$$

- The valve receives the information 288 times:
  Reception cost:

$$288*Pub\_Size * Erx = 288*(68*8)\ bit*58nJ/bit = 9.087mJ$$

Valve sends one SUBSCRIBE and receives one SUBACK:
Transmission cost:

$$Subscribe\_Size * Etx = (58*8)bit * 50nJ/bit = 0.0232mJ$$

Reception cost:

Sub_ACK_Size * Erx = (52*8)bit * 58nJ/bit = 0.024128mJ

- Valve average temperature computation (48 times):

48*2.4mJ = 115.2mJ

Total Energy Consumption:

0.0432+ 0.0436+7.8336+9.087+0.0232+0.0241+115.2 = 132.25mJ

EQ2:

Exercise Question 2 (EQ2): Propose **atleast one** solution for decreasing the energy consumption when passing using the Raspberry PI as a broker. **Give a rough estimate of the energy saving** that could be obtained with your solution: recompute the energy under your proposed configuration.

1. A potential solution to reduce energy consumption is to decrease the number of publish messages received by the valve.
   Currently, the valve receives a message every 5 minutes (288 messages per day), but it only computes the average temperature every 30 minutes (48 times per day).
   A more efficient configuration would be for the broker (Raspberry Pi) to receive and store the 5-minute temperature updates from the sensor, compute the average every 30 minutes, and forward only that average to the valve.
   The number of published messages received by the valve drops from 288 to 48.
   The valve no longer needs to process the average temperature, saving its 115.2 mJ processing cost.

   So instead of 288*Pub_Size * Erx we have

   48*Pub_Size * Erx = 48*(68*8)bit*58nJ/bit= 1.514mJ

   Here, we save 9.087 – 1.514 = 7.573 mJ

   Therefore, the new total energy consumption is:

   132.25 – 7.573 – 115.2 = 9.477mJ

   Of course, in this case, the energy consumed by the Raspberry Pi increases, since its role is no longer just to forward messages—it now also computes the average temperature. However, in this analysis, we are only considering the energy consumption of the sensor and the valve.

2. Another idea for reducing energy consumption is to use the MQTT-SN protocol. In MQTT-SN, packet sizes are generally smaller. However, the publisher must first send a Register Request and receive a Register Acknowledgment (ACK).

Since we don't have the exact packet sizes for MQTT-SN, we'll make the following assumptions:

- Register Request = 59 Bytes
- Register ACK = 51 Bytes
- PUBLISH packets are reduced by 10 Bytes compared to MQTT, i.e., from 68 Bytes → 58 Bytes

Transmission energy (Sensor → Broker):

288*Pub_Size * Etx = 288*(58*8)bit*50nJ/bit = 6.681mJ (instead of 7.8336mJ)

Reception energy (Broker → Valve):

288*Pub_Size * Erx = 288*(58*8) bit*58nJ/bit = 7.75mJ (instead of 9.087mJ)

We must consider that the sensor will send also a Register request and receive a register ack:

Reg_Size*Etx = (59*8)bit * 50nJ/bit = 0.0236mJ

Ack_Size * Erx = (51*8)bit * 58 nJ/bit = 0.02366mJ

Total Energy Saved: (7.8336-6.681) + (9.087 – 7.75) - 0.0236 - 0.02366 = 2.44234mJ

Total Energy Consumption with MQTT-SN: 132.25 – 2.44234 = 129.8 mJ

This is an estimation, as we're assuming a 10-byte reduction in PUBLISH packets. The actual savings may be higher or lower depending on the exact structure of MQTT-SN packets.

3. This idea is similar to the first one, but with a slight variation focused on optimizing data relevance rather than frequency.
   Since the sensor sends the temperature every 5 minutes, but in many cases the temperature may not change significantly within such a short interval, we can make the broker (Raspberry Pi) forward the temperature to the valve only when it differs from the last received value.
   This approach reduces the number of messages forwarded to the valve, while still maintaining meaningful data. To ensure the valve doesn't miss updates entirely, we enforce a minimum of one message every 30 minutes.
   The valve will then compute the average based on however many values it receives in each 30-minute interval—possibly fewer than 6 if there are duplicate temperature readings.