A Publish Subscribe based Middleware for Enabling Real Time Web Access on Constrained Device

Adhitya Bhawiyuga, Dany Primanita Kartikasari, Eko Sakti Pramukantoro
Faculty of Computer Science
University of Brawijaya
Malang, Republic of Indonesia
Email: bhawiyuga@ub.ac.id, dany.jalin@ub.ac.id, ekosakti@ub.ac.id

***Abstract*—In IoT world, web platform can be one of the promising interoperability enabler platform due to its massive success in current internet era. In order to integrate the web and IoT, a middleware with efficient communication architecture is required. In this paper, we designed and implemented publish subscribe based middleware for enabling real time web access on constrained device. The proposed system consists of three components : the sensor-actuator equipped device and its web based client counterpart acting as both publisher/subscriber, and MQTT broker. In order to send its sensing data to user, each device periodically publish a message with specific topic. In another side, user subscribe to that topic for getting the latest data from a device. The similar natures are applied for communication in reverse direction i.e. the user giving command to device. For enabling web access to the system, web based client utilize MQTT through websocket protocol to communicate with broker while the device utilize MQTT on top of pure TCP/IP stack to reduce the communication overhead. From the functional and performance testing we conclude that the proposed system is able to bridge the communication between web based client to the device with reasonable performance.**

I. Introduction

Communication among physically connected devices is one of important component in the perspective of Internet of Things (IoT). Therefore, many researchers from industry and academia have invented various protocols such as IEEE 802.11, IEEE 802.15.4, Bluetooth Low Energy (BLE), 6LowPan, CoAP, MQTT and so on [1]. While the goal of those protocols is for enabling the communication between heterogeneous devices, it may still requires an additional interoperability layer to provide an unified way to access those devices [3].

The web platform can be one of the promising interoperability enabler platform due to its massive success in current internet era[5]. Furthermore, by accommodating web standard into IoT world, it opens the possibility for IoT physical devices to be accessed by various kinds of client including web browser and mobile apps. Those two motivations now lead the transformation of IoT paradigm to the Web of Things (WoT) paradigm. In detail, the goal of WoT is to provide a mechanism so that the different physical devices can be treated in a similar way i.e. as a web resource[6].

Several proposed method for integrating the standard web into IoT world can be found on literature. The existing methods can be classified into two approaches: native HTTP-REST and REST-to-COAP. In first approach, authors propose the implementation of WoT by developing micro native HTTP server into sensor equipped device [10], [11], [12]. Each node can then be accessed directly through standard HTTP request. While it offers a simple design and architecture, the direct installation of HTTP server on a constrained device can be quiet expensive in term of computation and energy consumption. The second approach propose the middleware which translates the HTTP-REST into CoAP request [7], [8], [9]. Middleware first receives the request in standard HTTP which is then forwarded to the constrained device using CoAP protocol. In this case, middleware acts as both HTTP server and CoAP client. Even though the second approach moves the burden of being HTTP server from device to middleware, however, as the first approach, it still utilize the protocol that follow the request-response communication architecture i.e. HTTP and CoAP. In this case, the client e.x. web browser still needs to actively send the request to obtain latest data from the device which may not be suitable for real time communication purpose e.x. for streaming data collected by the device. Therefore, a communication protocol with more data oriented nature is required to cope with that challenge.

One of the popular application layer protocol in IoT world is Message Queue Telemetry Transport or simply called MQTT. As difference with HTTP, the MQTT is a TCP based messaging protocol with publish subscribe architecture. There exists three kind of actors in publish subscribe architecture : publisher, subscriber and broker [4]. Publisher sends the message identified by a specific topic to broker which then forwards that message to every subscriber who are interesting to that particular topic. Notice that, the publisher/subscriber can be either the IoT device or user who are willing to access that device. In this case, if an IoT device acting as publisher would like to multicast its sensor data to a specific group of user, it just need to send those data to broker with a specific topic. This kind of architecture is suitable for IoT case since it can provide a more data oriented protocol which can reduce the burden of a IoT constrained device for sending/receiving message.

This paper proposes a publish subscribe based middleware for enabling real time web access on constrained device. The proposed system consists of three components : the sensor-actuator equipped device and its web based client counterpart acting as both publisher/subscriber, and MQTT broker. In order to send its sensing data to user, each device periodically

978-1-5090-6477-9/17/$31.00 ©2017 IEEE

publish a message with specific topic. In another side, user subscribe to that topic for getting the latest data from a device. The similar natures are applied for communication in reverse direction i.e. the user giving command to device. For enabling web access to the system, web based client utilize MQTT through websocket protocol to communicate with broker while the device utilize MQTT on top of pure TCP/IP stack to reduce the communication overhead. From the functional and performance testing we conclude that the proposed system is able to bridge the communication between web based client to the device with reasonable performance.

The remainder of this paper is organized as follows. We first explain our preliminary study at section II. Then, in sections IV and V, we describe the design and implementation of our proposed system. The feasibility testing results are then discussed in section VI. Finally, we conclude our paper at section VIII.

II. Preliminary Study

A. Preliminary Study on Websocket

HTTP was designed for half duplex communication, where traffic flow in a single direction at one time. When the client sends a request to the server in one direction, then the server will responds to the request on the other direction. To solve the problem of inefficiency communication due to HTTP request-response mechanism, websocket make a new way for web communication to allow browser complete full duplex communication based on HTTP protocol [15]. The protocol was introduced in [16] developed as part of HTML5 standard. Websocket protocol work together with websocket API to perform websocket service. Many research have been conducted to develop web socket usage for real time application. One of them is published in [17] discussing about their web application called WindComm, aims to measure the one-way transmission latency of sending real-time wind sensor data.

Websocket protocol is similar in functionalities to TCP protocol. But it has backward compatibility to the existing web infrastructure. Fig. 1 shows sequence diagram of a websocket session. It needs to establish TCP connection between client and server before any websocket interaction begin to transmit data. After TCP connection established, web socket session following to the next step. Client transmits a websocket ugrade request to the server, and server send reply as a websocket upgrade response. Now websocket session is available, and both server and client may send data back and forth in asynchronous full duplex mode. [18].

B. Preliminary Study on MQTT

Publish/Subscribe paradigm is a communication for asynchronous messaging in distributed system. It described in [13] that P/S provide interaction schemes provides by two main entities: publishers and subscribers. First entity is role as data provider and the second entity is role as data consumers. Participant not only may act as publisher or as subscriber, but also may act as both roles. It depends on their necessity, whether they need to produce data or to gather data. In IoT system, there are many protocols exist for transfering data. MQTT is

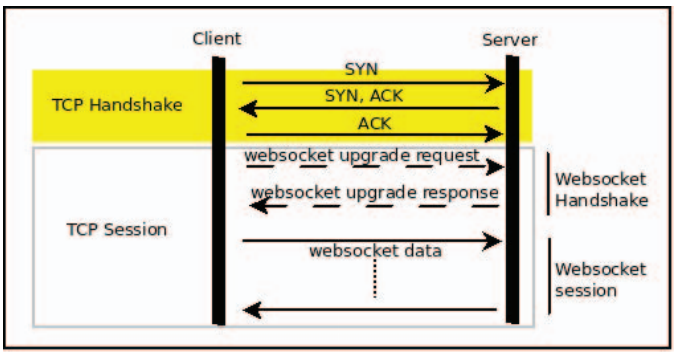

Fig. 1: Websocket over TCP sequence diagram.

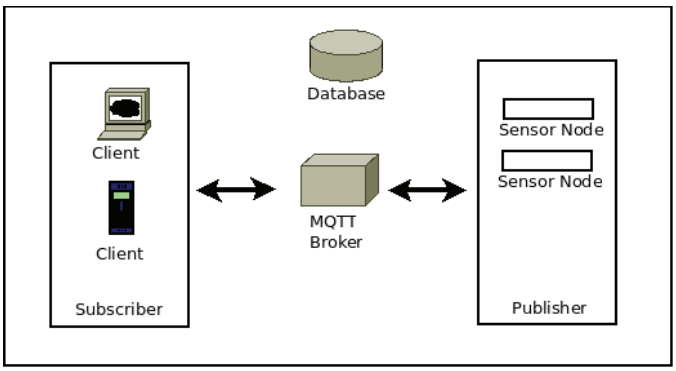

Fig. 2: Architecture of Publish Subscribe.

one of the most suitable for IoT implementation. MQTT is lightweigth protocol which suited for M2M communication and other remote application [14]. MQTT uses publish/subscribe mechanism, adopted client server architecture which connected by a device called broker. MQTT architecture is main communication point of the system. It responsible to lead the communication between clients connected. It has lightweight database to stores the data. Broker also responsible for dispatching message to the rightful subscribers. MQTT approach is illustrated in Fig 2.

III. Requirement Analysis

Given the background and problem stated in section I, the proposed system should satisfy following functional requirements :

1) The IoT device should be able to publish the sensor data with a specific topic.
2) The IoT device should be able to subscribe and receive message from web based client with a specific topic
3) The web based client should be able to publish the command to a specific IoT device by using a topic.
4) The web based client should be able to subscribe and receive sensor data from IoT device

IV. System Design

Fig. 3 illustrates the general architecture of our proposed system. There exists three main actors in this system :

1) the sensor-actuator equipped device as publisher/subscriber

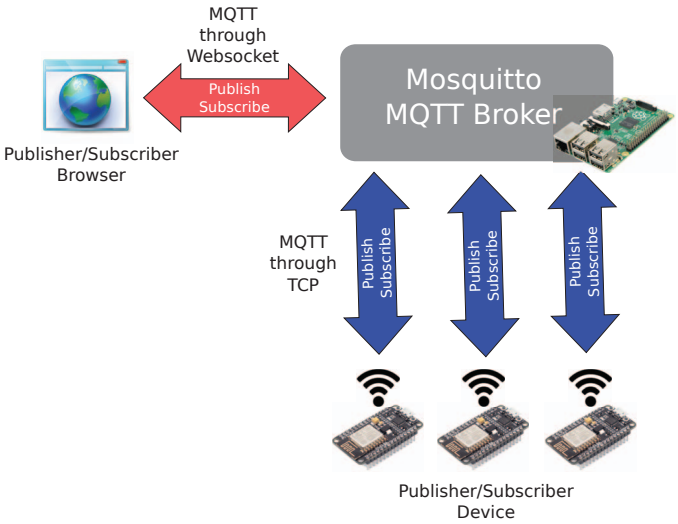

Fig. 3: General System Architecture.

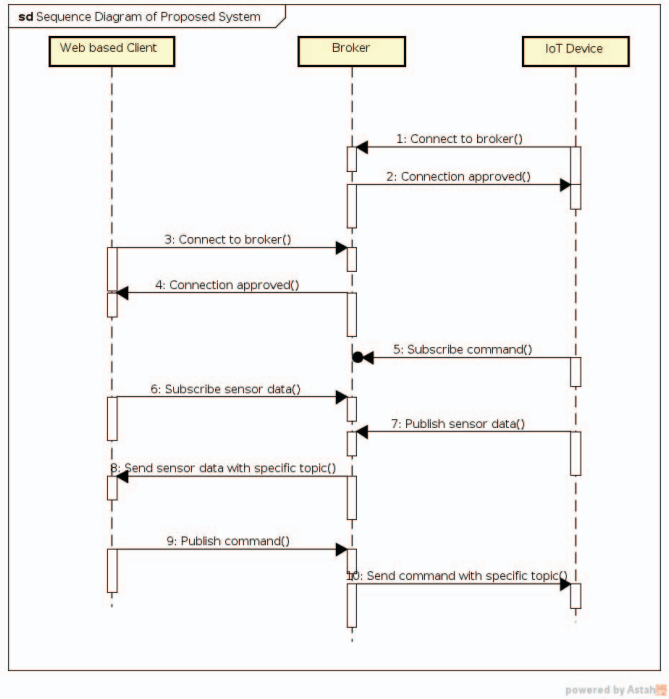

Fig. 4: Sequence Diagram of Proposed System.

2) web based client as publisher/subscriber
3) MQTT broker

In sensing scenario, the IoT device acts as publisher who get the measurement data from attached sensor and publish it to broker with a specific topic. In the other side, web based client aiming to get that data should subscribes to topic similar with the one published by IoT device. For example, if an IoT node containing two sensors : temperature and humidity can utilize following topics : **/node1/temperature** and **/node1/humidity**, respectively.

The similar natures are applied for communication in reverse direction i.e. the user giving command to device. The web based client send message containing a specific command

Listing 1: Payload of Command Message

```
{
    "act": "on",
    "dev": "lamp"
}
```

to the broker with a specific topic. The IoT device, in another side, subscribes to a specific topic to differentiate himself with other device. For example, if user would like to turn on the lamp connected to **node1**, he can send a command message with topic **/command/node1** which contains payload specified in Listing 1.

Notice that, for enabling web access to the system, web based client utilize MQTT through websocket protocol to communicate with broker while the device utilize MQTT on top of pure TCP/IP stack to reduce the communication overhead.

V. Implementation

In this section, we present the implementation of our proposed system in term of hardware and software design.

A. IoT Device as Publisher/Subscriber

We implement the prototype of IoT device using NodeMCU board. That hardware has following specification :

- Single-board microcontroller
- 128KB memory
- 4MB storage
- IEEE 802.11 b/g/n integrated communication module

In order to sense the environment we attach DHT-11 sensor aiming to get the temperature and humidity. For prototyping the actuator, we connect the board with relay switch to control a lamp.

For software part, we develop a Lua based program which is then installed on the board. The pseudocode of software part is presented in Listing 2. We first connect the device to MQTT broker with specific IP and port 1883. The device then subscribe to a specific topic i.e. **/node1** in order to receive the command from user. If the action is **on**, then device turn on the lamp by emitting signal 1 to a pin. For sending the sensor data, the device periodically publish message each second.

B. Web based Client as Publisher/Subscriber

For web based client part, we implement the logic part in Javascript using Paho MQTT Websocket JS Library. The main core of the library utilizes websocket as its transporting layer. Listings 3 shows the implementation pseudocode of this part.

C. MQTT Broker

For broker part, we utilize Mosquitto version 1.4.14 broker which has been configured to accept the websocket connection request. The configuration can be changed in **/etc/mosquitto/conf.d/websocket.conf**.

Listing 2: Pseudocode of IoT Device

```
m = mqtt.Client("m1", 120)
m:connect("IP", 1883, 0)
m:subscribe("/node1", 0)

function publish_data()
  temp=getTemp()
  m.publish("/node1/temp", temp, 0, 0)
end

function onMessage(client,topic,data)
  if data ~= nil then
    msg=json_parse(data)
    if msg["act"]=="on" then
        setDigital(PIN,on)
      else
        setDigital(PIN,off)
    end
  end
end

m:on("message",
  onMessage(client, topic, data))
tmr.alarm(2,1000,tmr.ALARM_AUTO
  ,publish_data)
```

Listing 3: Pseudocode of Web based Client

```
cl = new Paho.MQTT.Client(IP,Port,"ID");
cl.onMessageArrived = onMessageArrived;
cl.subscribe("/node1");
function onMessageArrived(message) {
  showMessage(message);
}
button.setOnClick(function(){
  cmd={"act":"on",
       "dev":"lamp"
      };
  cl.publish("/command/node1",cmd);
})
```

VI. Testing Scenario

VII. Result and Discussion

In this section, we present the testing of our proposed system. In general, we perform two testing : functional testing to measure the usability of proposed system and performance testing to measure the performance of proposed system.

A. Functional Testing

The functional testing is performed by running web based client, Mosquitto broker and IoT device. Fig. 5 and 6 shows the interface of web page for controlling and sensing, respectively. From Fig. 6 we can conclude that the web page is able to receive and show the sensor data published by IoT device.

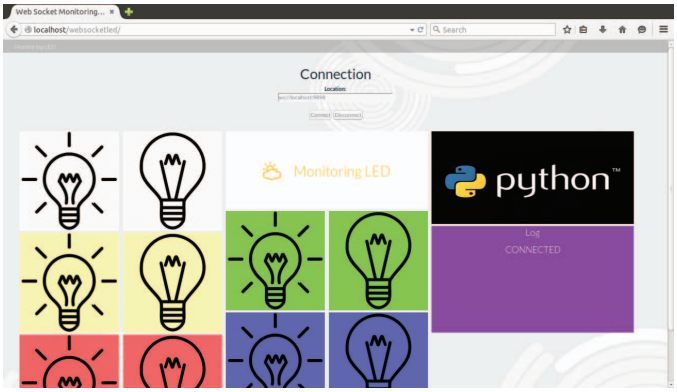

Fig. 5: Screenshot of Controlling Web Page.

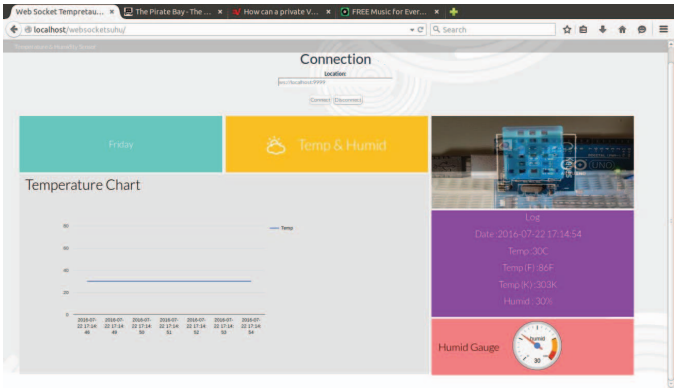

Fig. 6: Screenshot of Sensing Web Page.

Furthermore, from Fig. 5 we obtain that the proposed system is able to provide a control for device in the form of turning the light on or off.

B. Performance Testing

In order to obtain the performance of proposed system, we measure the delay from the time publisher send message until subscriber receive it in following scenarios : 1 publisher to N subscriber and N publisher to 1 subscriber. Fig. 7 and 8 shows the result of performance testing for 1-to-N and N-to-1 scenario respectively. From the graph we can see that there is an increasing trend of delay once the number of publisher increases and a relatively stable trend once the number of subscriber increases. However, on average, the message can still be delivered below 1s.

VIII. Conclusion

In this paper, we designed and implemented publish subscribe based middleware for enabling real time web access on constrained device. The proposed system consists of three components : the sensor-actuator equipped device and its web based client counterpart acting as both publisher/subscriber, and MQTT broker. For enabling web access to the system, web based client utilize MQTT through websocket protocol to communicate with broker while the device utilize MQTT on top of pure TCP/IP stack to reduce the communication overhead. From the functional and performance testing we

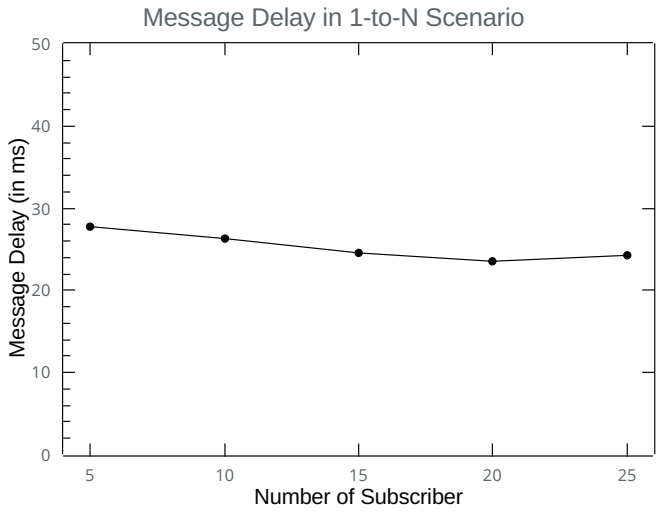

Fig. 7: Message Delay for 1-to-N Scenario.

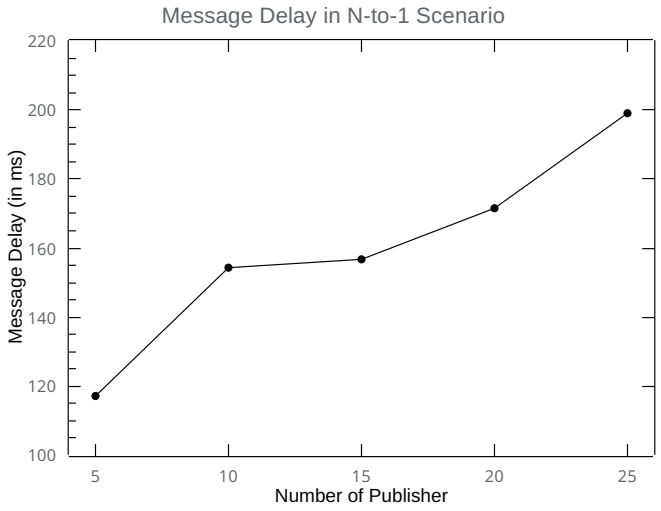

Fig. 8: Message Delay for N-to-1 Scenario.

conclude that the proposed system is able to bridge the communication between web based client to the device with reasonable performance.

ACKNOWLEDGEMENT

This work has been funded by Information Centric Networking Research Group (ICN-RG), Faculty of Computer Science, University of Brawijaya.

REFERENCES

[1] Atzori, L., Iera, A., & Morabito, G. (2010). The Internet of Thinags: A survey. Computer Networks, 54(15), 27872805.

[2] Granjal, J., Monteiro, E., & Sa Silva, J. (2015). Security for the Internet of Things: A Survey of Existing Protocols and Open Research Issues. IEEE Communications Surveys & Tutorials, 17(3), 12941312.

[3] Al-fuqaha, A., Member, S., Guizani, M., Mohammadi, M., & Member, S. (2015). Internet of Things : A Survey on Enabling, 17(4), 23472376.

[4] Banks, Andrew, and Rahul Gupta. "MQTT Version 3.1. 1." OASIS standard 29 (2014).

[5] Heuer, J., Hund, J., & Pfaff, O. (2015). Toward the web of things: Applying web technologies to the physical world. Computer, 48(5), 3442.

[6] Raggett, D. (2015). The Web of Things: Challenges and Opportunities. Computer, 48(5), 2632.

[7] Giang, N. K., Ha, M., & Kim, D. (2013). SCoAP: An integration of CoAP protocol with web-based application. GLOBECOM - IEEE Global Telecommunications Conference, 26482653.

[8] Mainetti, L., Mighali, V., & Patrono, L. (2015). A software architecture enabling the web of things. IEEE Internet of Things Journal, 2(6), 445454.

[9] Paganelli, F., Turchi, S., & Giuli, D. (2014). Web of things framework for RESTful applications and its experimentation in a smart city. IEEE Systems Journal, PP(99), 112.

[10] W. Drytkiewicz, I. Radusch, S. Arbanowski, and R. Popescu-Zeletin, "pREST: a REST-based protocol for pervasive systems," in *Proc. MAHSS*, 2004, pp. 340348.

[11] T. Luckenbach, P. Gober, S. Arbanowski, A. Kotsopoulos, and K. Kim, "TinyREST: A protocol for integrating sensor networks into the internet," in *Proc. REALWSN*, Stockholm, Sweden, 2005.

[12] V. Gupta, A. Poursohi, and P. Udupi, "Sensor.Network: An open data exchange for the web of things," in PERCOM Workshops, Mannheim, 2010, pp. 753755.

[13] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, The Many Faces of Publish/Subscribe, ACM Comput Surv, vol. 35, no. 2, pp. 114131, Jun. 2003.

[14] A. Sahadevan, D. Mathew, J. Mookathana, and B. A. Jose, An Offline Online Strategy for IoT Using MQTT, in 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud), 2017, pp. 369373.

[15] L. Yang and G. Yang, Real-Time Wireless Signal Testing and Analyzing System Based on Websocket, in 2016 Sixth International Conference on Instrumentation Measurement, Computer, Communication and Control (IMCCC), 2016, pp. 648652.

[16] I. F. ¡ifette+ietf@google.com¿, The WebSocket Protocol. [Online]. Available: https://tools.ietf.org/html/rfc6455. [Accessed: 02-Aug-2017].

[17] V. Pimentel and B. G. Nickerson, Communicating and Displaying Real-Time Data with WebSocket, IEEE Internet Comput., vol. 16, no. 4, pp. 4553, Jul. 2012.

[18] D. Skvorc, M. Horvat, and S. Srbljic, Performance evaluation of Websocket protocol for implementation of full-duplex web streams, in 2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2014, pp. 10031008.