Efficiency of integration between sensor networks and clouds

Original Scientific Paper

Filip Tsvetanov
South West University,
Faculty of Engineering, Department of Communication and computer engineering
Ivan Mihajlov street 56, Blagoevgrad, Bulgaria
ftsvetanov@swu.bg

Martin Pandurski
South-West University,
Faculty of Engineering, Department of Communication and computer engineering
Ivan Mihajlov street 56, Blagoevgrad, Bulgaria
tom1000@abv.bg

Abstract *– Numerous wireless sensor networks (WSN) applications include monitoring and controlling various conditions in the environment, industry, healthcare, medicine, military affairs, agriculture, etc. The life of sensor nodes largely depends on the power supply type, communication ability, energy storage capacity and energy management mechanisms. The collection and transmission of sensor data streams from sensor nodes lead to the depletion of their energy. At the same time, the storage and processing of this data require significant hardware resources. Integration between clouds and sensor networks is an ideal solution to the limited computing power of sensor networks, data storage and processing. One of the main challenges facing systems engineers is to choose the appropriate protocol for integrating sensor data into the cloud structure, taking into account specific system requirements. This paper presents an experimental study on the effectiveness of integration between sensor networks and the cloud, implemented through three protocols HTTP, MQTT and MQTT-SN. A model for studying the integration of sensor network - Cloud with the communication models for integration - request-response and publish-subscribe, implemented with HTTP, MQTT and MQTT-SN. The influence of the number of transmitted data packets from physical sensors to the cloud on the transmitted data delay to the cloud, the CPU and memory load was studied. After evaluating the results of sensor network and cloud integration experiments, the MQTT protocol is the most efficient in terms of data rate and power consumption.*

Keywords*: communication protocols, integration, sensor networks, cloud.*

1. INTRODUCTION

Modern technological advances in sensor architecture, device miniaturisation, and wireless networks have facilitated wireless sensor networks' design, distribution, and application (WSN). The sensor networks are self-organising and consist of many different types of sensors, equipped with tools for monitoring, processing and communication, which are located in a certain area for monitoring, controlling and transmitting data to each other via wireless communication. The applications of WSN are numerous and include monitoring and control of a wide variety of conditions in the environment, everyday life, industry, healthcare, medicine, military affairs, agriculture, etc. The life of sensor units largely depends on the power supply types, their ability to communicate, energy storage capacity and energy management mechanisms. The collection and transmission of sensor data from sensor nodes lead to the depletion of their energy. At the same time, the storage and processing of this data require significant hardware resources. Designing additional capability to process the collected data can significantly increase the cost of the sensor. Due to the lack of battery power and bandwidth, the sensor nodes cannot store and process extensive data [1]. Therefore, storing and processing raw data is a challenging task.

On the other hand, cloud structures provide enormous computing power and storage space. Integration between clouds and sensor networks is an ideal solution to the limited computing power of sensor networks, data storage and processing. A new paradigm called "Sensor Cloud Computing" has been formulated to achieve this integration. Therefore, the sensor cloud arises to perform many tasks that are not possible from sensor networks [2]. The widespread use of WSN in many processes poses more and more severe problems related to the ability of people to share and analyse

sensor data in real-time. This large volume of data is a prerequisite for the trend for many companies to prefer and switch to using cloud databases for data storage and processing. Therefore, the collected sensory data is not only stored and processed in the clouds but can be accessed anywhere, anytime. Maintaining and providing the resources to the end-users of the sensor cloud is a challenging and important task. Researchers from academia, industry and standards organisations continue to work and offer potential solutions to this challenge.

This paper presents the sensor cloud architecture, focusing on the sensor network-cloud integration process. Some of the most used integration protocols are briefly analysed. The goal is to conduct an integration efficiency implementation experiment with an actual physically built sensor network. That network will send data to the cloud and performs tests on the influence of the different parameters that transmit data packets via HTTP (Hypertext Transfer Protocol), MQTT (message queue telemetry transport) and MQTT-SN (sensor network) protocols.

The paper contains a representation of Communication protocols for sensor data integration, the impact of the protocols on the integration effectiveness, Models for studying the integration of sensor data into a cloud structure, Communication models for sensor network - cloud interaction, Experimental design, Experimental study of the parameters influence of the transmitted packets on the delay, Results and discussions, Conclusion.

2. LITERATURE REVIEW

The conclusions that the authors give in [4, 5, 8, 13, 14, 15,16,17] can be systematised like this: MQTT is more suitable over HTTP when the same connection is reused as much as possible. If connections are created and broken frequently to send individual messages, the performance is not considerable compared to HTTP.

Except the protocols message format, another important feature, determining the integration efficiency, is the protocols communication models.

MQTT uses Pub/Sub model with broker, which collect all data and sends particular messages, only to clients, that are subscribed for them. In this way the payload is reduced, and so it is better for WSN than HTTP Request/Response model [14,15,16,17,18,19,20].

The proposed methods [13,14,15,16,17,18,19,20,21] cannot be easily applied in many IoT applications due to the limitations of IoT devices. Depending on the functional requirements of each model, a suitable solution would be using gateway devices/software with higher processing/memory capabilities. The data is transmitted from end devices to the Gateway, where various optimisation methods can be applied before further transmissions to the cloud [14,15,16,17,18,19,20,21,22].

3. COMMUNICATION PROTOCOLS FOR INTEGRATION OF SENSOR DATA TO CLOUD

The biggest challenge in designing "sensor-cloud" systems is establishing a communication channel between devices, gateways, servers and cloud platforms. Therefore, this task requires the use of different protocols. The complete communication stack contains the protocols distributed in four different layers: application, transport, Internet and the channel layer [3]. Some characteristics of popular protocols for "sensor-cloud" integration are shown in Table 1.

Table1. Protocols for integration of WSN into the cloud

Protocols	Characteristic			
	Communication model	Transport layer	QoS	Security
HTTP	Request - response	TCP	-	TLS/SSL
MQTT	Publish-subscribe	TCP	QoS-0, QoS-1 QoS-2	TLS/SSL
MQTT-SN	Publish-subscribe	UDP	QoS-0, QoS-1 QoS-2	TLS

As can be seen from Table 1. the communication channel can be established by appropriate data transmission protocols. By selecting a protocol in the application layer, we can influence the settings in the transport and Internet layers protocols to be predefined. The channel layer is usually determined by the hardware solutions, including IEEE 802.15.4, Z- wave, 802.11 WiFi, Bluetooth Low Energy (BLE), Zigbee, etc.

In sensor and IoT (Internet of things) networks, many small data blocks from different devices are transferred across different networks. Although the Internet Protocol IP is accepted for most types of communication, it has some problems when applied to IoT sensor networks. Internet access requires application protocols running over TCP/IP (Transmission Control Protocol) or UDP/IP (User Datagram Protocol). In addition, IP addressing depends on the physical location, which causes the problem of network control complexity. To address these issues, various name-based architectures have been discussed, such as Named Data Networking (NDN), Content-Centric Networking (CCN), and Information-Centric Networking (ICN) [4], [5], [6]. MQTT is one of the most commonly used protocols in name-based architectures because it reduces high data transmission costs and provides highly efficient communication in IoT systems. It also uses Name-based routing, thus reducing the need for routing, compared to IP addresses, for IoT traffic flows.

3.1. HTTP FOR COMMUNICATION IN SENSOR NETWORKS

HTTP determines how messages are transmitted and formatted on the Internet and all websites. HTTP trans-

fers many small packets when communicating with sensor and IoT devices and provides reliable communication over TCP/IP. Connections established by TCP are released on each access, as the available data is transferred based on IP and URL address and their connection changes dynamically [7]. This communication feature in sensor and IoT devices causes severe costs and consumption of network resources, and long delays.

3.2. MQTT

MQTT is ideal for use in many situations, including limited environments, such as communication in M2M and IoT, requiring low power consumption, making it one of the most popular protocol solutions for data transmission in the limited environments [8]. The protocol works on TCP/IP, providing orderly, lossless two-way connections. The MQTT Publish/Subscribe paradigm is event-driven and allows messages to be moved between a broker and two MQTT clients (publisher/subscriber). The broker receives and processes all messages, separates the publisher from the subscriber and acts as a router for the messages, deciding where to send them [9]. The publisher, in turn, creates different topics in the broker, as shown in Fig.1.

The MQTT has three different levels of Quality of Service QoS 0, QoS 1 and QoS 2. The QoS level determines the delivery guarantee of a specific message.

MQTT offers SSL/TLS protocols and a client SSL certificate for the security of the transferred content. The MQTT protocol is not text-based, and without SSL/TLS, communication is fully open, and the password is the main concern.

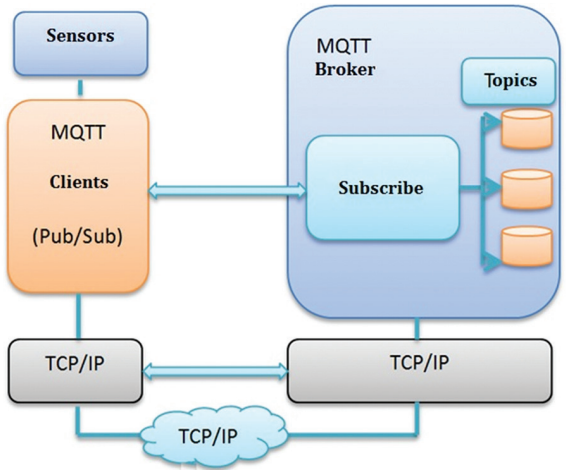

Fig.1. MQTT Architecture

Depending on the desired level of security, the MQTT protocol prescribes the following TCP channels:

- 1883 = non-encrypted MQTT and the channel should not be used for sensitive data.
- 8883 = encrypted MQTT, data is encrypted with SSL/TLS, and customer support is required to establish the connection.
- 8884 = encrypted MQTT + client certificate - this is the highest level of security available for MQTT communication. In addition to the encrypted data using the SSL/TLS protocol, the client must authenticate with a certificate issued by the broker. So far, however, this channel is maintained by only a few public brokers (e.g. Mosquitto - test.mosquitto.org server).

3.3. MQTT-SN

MQTT-SN is an adapted version of MQTT for WSN, making it suitable for sensor devices due to its low power, bandwidth limitation, and compact messaging. MQTT-SN uses UDP/IP transport communication protocol because it's lighter than TCP/IP. There are three types of MQTT-SN components: MQTT-SN clients, MQTT-SN GW gateways, and MQTT-SN forwarders [10].

MQTT-SN clients connect to the MQTT broker/server via the MQTT-SN GW using the MQTT-SN protocol. MQTT-SN forwarders are responsible for transporting messages to GW. The gateways used are of two types [11], [12]:

- *Transparent Gateway*, where each MQTT-SN connection has a corresponding MQTT connection. This is the most accessible type to implement.
- *The aggregating Gateway* represents multiple MQTT-SN connections that share a single MQTT connection.

4. INVESTIGATION OF THE IMPACT OF THE PROTOCOLS ON THE EFFECTIVENESS OF INTEGRATION

WSN faces many limitations and challenges related to the storage of large volumes of sensor data, their processing, scalability, security, accessibility, etc. Connecting the sensor network to the cloud structure solves the problem of storing, processing and transmitting large volumes of data generated by the sensor networks in real-time. This paradigm is known as "Sensor-Cloud" and can be implemented with physical and virtual sensors. Several advantages of using a "Sensor-Cloud" are described in [13]. The WSN - cloud communication can be realised through Gateway devices.

This study aims to assess the integration by examining the impact of the protocols type on the integration of sensor network data to Cloud - HTTP, MQTT and MQTT-SN. The integration evaluation can be done according to package number, topics per packet, and bit value criteria. The general requirement is reliable data transmission from sensor nodes to the database in the cloud. As a parameter's efficiency for the integration, we accept the delay of the transmitted data, the CPU (central processing unit) and RAM (random-access memory) load, showing the consumed energy degree.

4.1 MODEL FOR STUDYING THE INTEGRATION OF SENSOR DATA INTO A CLOUD STRUCTURE

The model from fig. 2 is in accordance with the scheme for data transmission between the sensor network and the Cloud via Gateway, analysed in detail in [14].

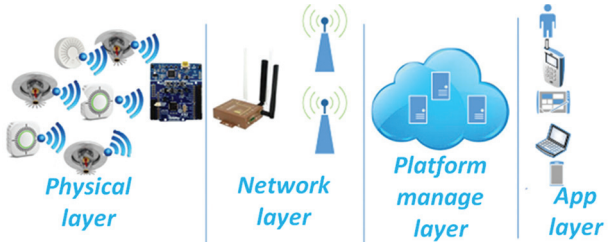

Fig. 2. Model for integrating sensor data to a cloud structure

The physical layer-includes different intelligent sensors that send data to the microcomputer.

The network layer includes a microcomputer that acts as a gateway and forwards data to a base station. The base station transports the received data to the cloud platform, which contains data storage and processing servers.

The platform management layer provides data storage and device management features.

The application service layer is connected to the cloud platform via the API (Application Programming Interface) to implement the function for online data requests and remote monitoring.

The advantage of the technology used is the possibility of remote control via mobile phone or tablet and the low cost.

4.2. COMMUNICATION MODELS FOR INTERACTION BETWEEN THE SENSOR NETWORK AND THE CLOUD STRUCTURE

The experiments were conducted with two communication models: "request-response" and "publish-subscribe". The communication model request-response is implemented with the HTTP protocol and publish/subscribe via MQTT and MQTT-SN.

4.3. DESIGN EXPERIMENT

The Experimental design includes

- An Xbee/Zigbee sensor network has been built.
- The sensor data is collected (via Routers in Mesh topology) and aggregated in the Coordinator.
- Then it's transmitted to the RPI4 microcomputer, which loads pre-developed code for the experiment. RPI4 transmits the sensor data to the ThingBoard Cloud [15] via MQTT, HTTP and MQTT-SN, Fig.4.

The ThingBoard Cloud is free code and supports various integration protocols. As can be seen from the documentation [15], the ThingBoard Cloud is not designed to access MQTT-SN data.

Fig. 3. Experiment design

MQTT-SN requires MQTT-SN Gateway, which acts as a protocol converter to convert MQTT-SN messages to MQTT messages [12].

4.4. EXPERIMENTAL STUDY OF THE INFLUENCE OF THE PARAMETERS FOR THE TRANSMITTED PACKETS ON THE DELAY.

The main focus of the proposed experiment is the influence of different parameters, such as number of packets, number of topics in packets and bits for each topic, on the speed (delay) of data transmission to the cloud via different protocols HTTP, MQTT and MQTT-SN. The parameter values can be changed via code settings. Many scenarios have been studied.

5. RESULTS AND DISCUSSIONS

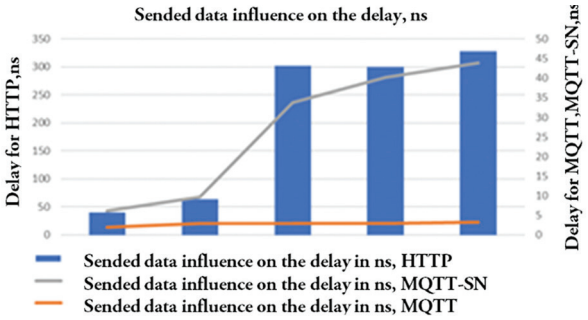

Fig.4. The number of transmitted packets Influences the delay

- A study of the number of transmitted packets influences the delivered data delay, Figure 4.

The study shows that we have the highest level of delay in HTTP due to the bigger header. Next in line is MQTT-SN due to gateway usage. MQTT offers a minor delay, a due smallest header of 2 bytes.

- A study of the number of transmitted packets influences RAM in MB

With packages increasing, the difference in protocol's impact on RAM load increases, in a way that HTTP shows the highest level of RAM stress, followed by MQTT-SN and MQTT. The conclusions of the results are based on the already discussed protocol's features.

- Number of transmitted packets Influence on CPU, MHz, Fig. 6.

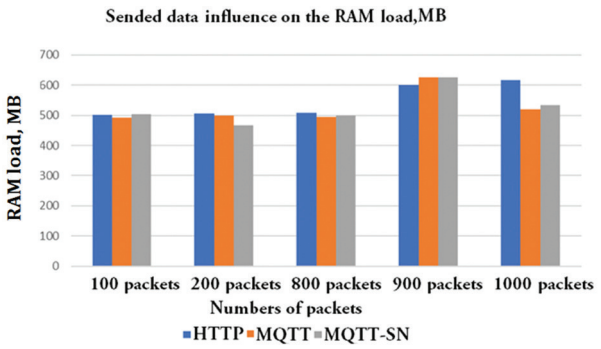

Fig. 5. Number of transmitted packets Influence on RAM

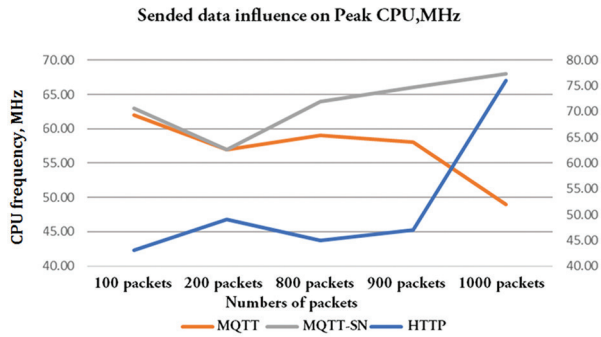

Fig. 6. Influence of the number of transmitted packets on the CPU

MQTT-SN shows the highest CPU load at the greatest packet numbers, followed by HTTP and the lowest load levels in MQTT. The results are based on the already discussed protocol's features.

- A study of the complex parameters influence the packets data delay

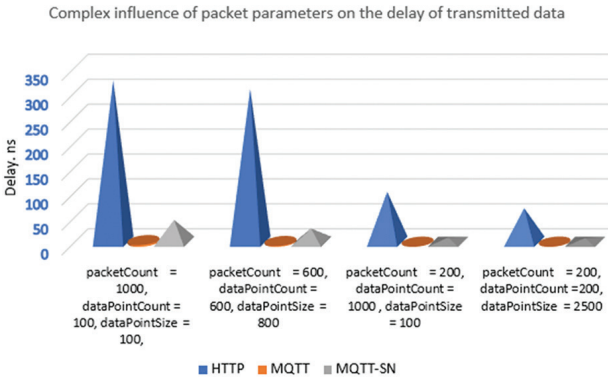

Fig. 7. The complex influence of packet parameters on the delay of transmitted data

This study considers the simultaneous influence of the three package parameters: number of packages, number of topics in the packages and bit value for each topic. Fig. 7 shows that in MQTT, the delay is almost unchanged, and MQTT-SN is in second place with minimal delay impact. It can be summarised that the biggest delay is in HTTP.

The limitations are on half of the ThingsBoard Cloud, which allows us to upload only a certain amount of data. Above that border, the cloud doesn't allow us to send more data.

- Verification of the transmitted data

The Wireshark software research tool performs the transmitted packets' destination and size verification for each examined protocol. Fig.8, fig 9 and fig.10 show the transmitted data via the HTTP, MQTT and MQTT-SN protocols.

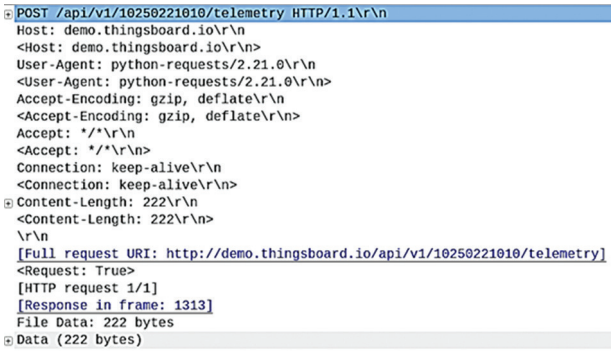

Fig. 8. HTTP POST commands

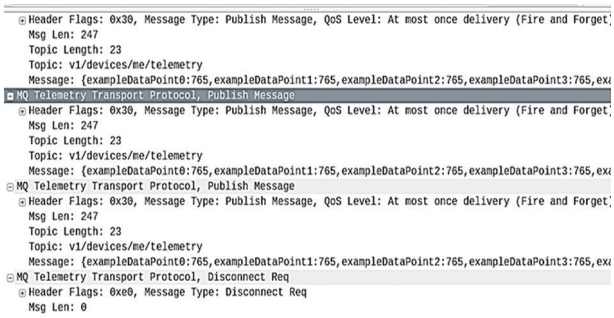

Fig. 9. MQTT PUBLISH and DISCONNECT commands

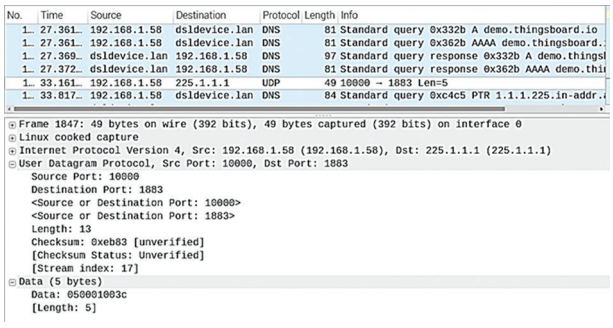

Fig.10. Send data via UTP for MQTT-SN

The analysis of the results, obtained from the experimental studies of the transmitted sensory data from physical sensors, gives grounds to draw the following conclusions:

MQTT provides the least delay in data transmission, the least CPU and RAM load, respectively, and requires

the least power consumption and the shortest time for data transmission.

MQTT-SN works with more significant delay and higher energy consumption. One of the reasons, in our opinion, is the additional processing that is implemented in the serialisation process using the MQTT-SN Gateway to convert data structures or objects to a byte stream, which explains the more significant delay.

Evaluating the results of integration experiments, the most efficient data rate and the energy consumption is MQTT.

The obtained results provide useful and practically applicable information for the designers of such systems on the efficiency of the transmitted data through the protocols for integrating sensor data HTTP, MQTT and MQTT-SN to the cloud structure.

The main contributions are the developed Python code for the experiments and the Gateway configurations in both hardware and software. In the general sense, Gateway converts different protocols at different levels. We can connect different devices/programs using gateways, working on different technologies, on single personally designed model platforms.

We plan to include more protocols, such as CoAP, because it works on UDP but uses a "request-response" model. Also, we plan to encrypt the protocols.

6. CONCLUSION

Integration between clouds and sensor networks is ideal for the limited computing power of sensor networks, storage, processing and access to sensor data anywhere and anytime. Designing, maintaining and providing end-user resources from the sensor cloud is a challenging and important task.

This paper is devoted to studying the integration between the sensor network and the cloud in transmitting sensor data to the cloud.

Has been created a model for studying the integration of sensor network - Cloud with the communication models for integration – "request-response" and "publish-subscribe", implemented with HTTP, MQTT and MQTT-SN.

An algorithm and Python code have been developed to conduct the experiments, the operability of which has been verified using the Wireshark tool.

The influence of the number of transmitted data packets from physical sensors to the cloud on the speed (delay) of the transmitted data to the Cloud, CPU and memory load was studied.

From evaluating the results of the experiments for integration between the sensor network and the cloud, the MQTT protocol is the most efficient in terms of data transfer rate and energy consumption.

7. REFERENCES:

[1] M. Pandurski, F. Tsvetanov, "Research of energy resources of the cluster sensor network ", Journal of Engineering Science and Technology Review, Vol. 13, 2020, pp. 32-36.

[2] F. A. Tsvetanov, "Storing Data from Sensors networks ", Journal IOP Conference Series: Materials Science and Engineering, Vol. 1032, 2021, p. 012012.

[3] M. Burhan, R. Rehman, B. Khan, B. Kim, "IoT Elements, Layered Architectures and Security Issues: A Comprehensive Survey", Sensors, Vol. 9, 2018, pp. 1-34.

[4] T. Yokotani, S. Yuya. "Comparison with HTTP and MQTT on required network resources for IoT", Proceedings of the International Conference on Control, Electronics, Renewable Energy and Communications, Bandung, Indonesia, 13-15 September 2016, pp. 1-6.

[5] Y. Sasaki, T. Yokotani, "Performance Evaluation of MQTT as a Communication Protocol for IoT and Prototyping", Advanced Technology Innovations, Vol. 4, No. 1, 2019, pp. 21-29.

[6] A. Viswanathan, "Analysis of Power Consumption of the MQTT Protocol", https://www.semanticscholar.org/paper/Analysis-of-Power-Consumption-of-the-MQTT-Protocol-Viswanathan/f7c4c44eeb9ed5b2ec465ff35f78352e696c0724 (accessed: 2022)

[7] C. R. Garzon, "HTTP Request Methods – Get vs Put vs Post Explained with Code Examples", https://www.freecodecamp.org/news/http-request-methods-explained (accessed: 2022)

[8] J. Dizdarević, F. Carpio, A. Jukan, X. Masip-Bruin. "A Survey of Communication Protocols for Internet of Things and Related Challenges of Fog and Cloud Computing Integration", ACM Computing Surveys, Vol. 1, 2018, 30 pages.

[9] Practical guide to MQTT and Mosquitto 2021, How to Install The Mosquitto MQTT Broker on Windows, http://www.steves-internet-guide.com/install-mosquitto-broker (accessed: 2022).

[10] C.-S Park, H.-M. Nam, "Security Architecture and Protocols for Secure MQTT-SN", IEEE Access, Vol. 8, 2020, pp. 226422-226436.

[11] B. Schütz, J. Bauer, N. Aschenbruck, "Improving Energy Efficiency of MQTT-SN in Lossy Environments Using Seed-Based Network Coding", Proceedings of the IEEE 42nd Conference on Local Computer Networks, Singapore, 9-12 October 2017, pp. 286-293.

[12] A. Stanford-Clark, H. L. Truong, "MQTT for sensor networks (MQTT-SN) protocol specification version 1.2.IBM", http://mqtt.org/new/wp-content/uploads/2009/06/MQTT-SN spec v1.2.pdf (accessed: 2022)

[13] D. R. Kumar, S. Si. Kumar., "Integration of Wireless Sensor Networks with Cloud: A Review", Proceedings of the 9th International Conference on Cloud Computing, Data Science & Engineering Confluence, Noida, India, 10-11 January 2019, pp. 114–119.

[14] F. Tsvetanov, M. Pandurski, "Some aspects for the integration of sensor networks in cloud structures", Proceedings of International Conference on High Technology for Sustainable Development HiTech , 2018, pp. 249-253.

[15] ThingsBoard Cloud Documentation https://thingsboard.io/docs/paas, (accessed: 2022)

[16] M. Collina, G. Corazza, "Introducing the QEST broker: Scaling the IoT by bridging MQTT and REST", Proceedings of the 23rd Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, Sydney, NSW, Australia, 9-12 September 2012.

[17] T. Yokotani, Y. Sasaki, "Comparison with HTTP and MQTT on Required Network Resources for IoT", Proceedings of the International Conference on Control, Electronics, Renewable Energy and Communications, Bandung, Indonesia, 13-15 September 2016.

[18] J. Dizdarevic, F. Carpio, "A Survey of Communication Protocols for Internet of Things and Related Challenges of Fog and Cloud Computing Integration", ACM Computing Surveys, Vol. 1, No. 1, 2018.

[19] Y. Sasaki, T. Yokotani, "Performance Evaluation of MQTT as a Communication Protocol for IoT and Prototyping", Advances in Technology Innovation, Vol. 4, No. 1, 2019, pp. 21-29.

[20] M. O. Reddy, J. Seventline, "Performance Analysis of QoS for the MQTT-SN Protocol with Industry Oriented MQTT-SN Gateway and Integration with Cloud MQTT-Server",. International Journal of Future Generation Communication and Networking Vol. 13, No. 3, 2020, pp. 2651-2673.

[21] F. Tsvetanov, M. Pandurski, "Security of the sensory data in the cloud", IOP Conference Series: Materials Science and Engineering, Vol. 1032, 2020, p. 012005.

[22] E. Davis, "Pub/Sub Protocol in WSN: Improved Reliability and Timeliness", Transactions on Internet and Information Systems Vol. 12, No. 4, 2018.