ADVANCES IN CYBER-PHYSICAL SYSTEMS

Vol. 10, No. 2, 2025

MQTT LATENCY EVALUATION IN CLOUD-BASED SPECTROMETER CONTROL

Andriy Krupych[1], Edgars Elsts[2]

[1]*Ivan Franko National University of Lviv, 1, Universytetska str., Lviv, 79000, Ukraine,*
[2]*Institute of Solid State Physics, University of Latvia, 8, Kengaraga, Riga, LV-1063, Latvia.*
Authors' e-mails*: andriy.krupych@lnu.edu.ua*, *edgars.elsts@cfi.lu.lv*

https://doi.org/10.23939/acps2025.02.151

Submitted on 30.09.2025

© Krupych A., Elsts E., 2025

***Abstract*: This paper investigates latency characteristics of MQTT communication in remote spectrometer control. A comparative study of AWS IoT Core and HiveMQ broker implementations across various Quality of Service levels has been presented. Methodologies include detailed measurements of control command and data feedback latencies, followed by statistical analysis. Initial findings have demonstrated a monotonic increase in latency and its variability (jitter) as QoS levels rise for both brokers, confirming the inherent MQTT trade-off between reliability and speed. AWS IoT Core consistently exhibits lower modal latencies and more concentrated distributions across all QoS levels compared to HiveMQ, suggesting superior average performance and consistency. The presented analysis has provided insights into how broker choice and QoS configuration impact remote control performance. Results can be useful for the development of more reliable, low-latency, and efficient remote laboratory systems for advanced scientific experimentation.**

***Index terms*: MQTT, AWS IoT Core, HiveMQ, QoS, latency.**

I. INTRODUCTION

This study undertakes a rigorous investigation into the latency characteristics of Message Queuing Telemetry Transport (MQTT) in the context of remote spectrometer control, specifically comparing AWS IoT Core and cloud-based HiveMQ broker implementations.

The current document emphasizes the importance of measuring MQTT latency for remote spectrometer control, as low latency is crucial for responsive and precise operation of remote laboratory equipment. Reliable message delivery is equally vital to ensure that commands are accurately transmitted to instruments and that critical feedback data is received without loss, enabling effective and safe experimental execution.

This work builds upon prior research titled "Computerized Optical Experiments with Portable Fiber Optic Spectrometers and Amazon Web Services Cloud Integration." [20] It expands upon the foundational insights of that previous study by specifically incorporating a detailed quality of service benchmark and a comprehensive latency profile of Message Queuing Telemetry Transport communication, further enhancing the understanding of remote spectrometer control performance.

Both AWS IoT Core and cloud-based HiveMQ brokers are being compared for their MQTT latency in remote spectrometer control. AWS IoT Core offers a fully managed, highly available environment with built-in scalability and robust security, ideal for low-latency, bi-directional communication. HiveMQ is noted for its scalability, reliability, efficient message handling via a cluster-based architecture, and strong security through TLS encryption, providing robust Quality of Service levels. The study aims to specifically benchmark these characteristics in the context of remote laboratory applications.

II. LITERATURE REVIEW AND PROBLEM STATEMENT

In the context of remote laboratory functioning, the importance of low latency and reliable message delivery cannot be overstated. Low latency is crucial for real-time control, ensuring that commands from a remote user are executed promptly by the equipment, and that feedback is returned without significant delay, which is essential for precise operation and preventing experimental errors [2]. Reliable message delivery guarantees that all commands and data packets reach their intended destinations, averting data loss or misinterpretation that could compromise experimental integrity or safety [8]. The combination of minimal delay and guaranteed message transfer directly impacts the accuracy, safety, and overall responsiveness of remote experimental setups.

MQTT incorporates Quality of Service levels (0, 1, and 2) to manage message delivery guarantees, allowing developers to balance between reliability and performance based on application requirements [3]. QoS 0 offers "fire and forget" delivery with no guarantee, QoS 1 ensures "at least once" delivery, and QoS 2 provides "exactly once" delivery [14]. Numerous studies have benchmarked MQTT QoS in various IoT applications, investigating its impact on throughput, latency, and energy consumption [3, 5, 9]. These benchmarks are critical for optimizing system performance and ensuring that the chosen QoS level aligns with the specific demands of remote laboratory operations, where data integrity can be paramount.

Benchmarking the performance of MQTT brokers is essential for selecting the most appropriate solution for specific IoT architectures. Research has focused on evaluating key metrics such as message exchange latency, offset response time, throughput, and CPU usage under various network conditions including delay, variance, and packet loss [2]. For instance, comparative analyses often involve popular open-source brokers like Mosquitto, EMQX, RabbitMQ, VerneMQ, and HiveMQ, providing insights into their respective strengths and weaknesses in edge computing contexts [2, 8]. Such evaluations help in understanding how different broker implementations handle varying loads and network complexities.

Among the commercially available MQTT brokers, HiveMQ is recognized for its enterprise-grade features, including high scalability, reliability, and robust security mechanisms [7]. Its cluster-based architecture is designed to handle a large number of connected devices and messages efficiently, offering Quality of Service levels that ensure dependable data exchanges [7]. Security is paramount, with HiveMQ implementing Transport Layer Security encryption and multiple authentication methods to protect sensitive data, making it suitable for demanding applications like agritech and industrial IoT [7].

Similarly, AWS IoT Core provides a fully managed and highly available cloud service designed to enable secure and efficient interaction between connected devices and cloud applications [1, 11]. It supports MQTT and offers robust security through authorization, access control, and TLS encryption for all traffic [1]. AWS IoT Core is inherently scalable, designed for low-latency, bi-directional communication, and integrates seamlessly with the broader AWS ecosystem, allowing for advanced data processing, analytics, and device management via services like Rules Engine, AWS Lambda, and various database systems [1, 4, 12, 18].

The application of these communication technologies extends to advanced scientific instrumentation, particularly in optical experiments. The development of portable fiber optic spectrometers and their integration with cloud services has enabled new possibilities for remote chemical analysis and spectroscopic measurements [13, 15]. Efforts to miniaturize spectrometers and connect them to cloud-based infrastructures have been driven by the need for in situ, in vitro, and in vivo measurements across diverse fields, offering cost-effective and accessible solutions for researchers and educators [10, 17, 19]. The foundational work on [20] laid the groundwork for integrating such instruments with cloud platforms.

Building upon this prior research, the current work aims to deepen the understanding of communication performance in remote optical experiments. By specifically incorporating a detailed quality of service benchmark and a comprehensive latency profile of MQTT communication between portable fiber optic spectrometers and both AWS IoT Core and cloud-based HiveMQ broker implementations, this study seeks to provide critical insights into optimizing remote control performance. This focused analysis will contribute to the ongoing development of reliable, low-latency, and efficient remote laboratory systems, thereby enhancing the capabilities and accessibility of advanced scientific experimentation.

III. SCOPE OF WORK AND OBJECTIVES

The remote laboratory framework integrates physical light-control and measurement devices with cloud-based interfaces through an IoT architecture. At the device level, a Latte Panda Single Board Computer connects a tunable RGB LED (controlled by an Arduino Leonardo) and a StellarNet spectrometer. A Python script on the SBC facilitates hardware-cloud communication: it receives control commands via MQTT to operate the LED, acquires spectral data from the spectrometer, and then publishes this processed data to the cloud. Users can then interact with the system through a web-accessible, real-time cloud dashboard, enabling both control of the light source and observation of the spectral feedback.

This study is focused on the comparative analysis of Message Queuing Telemetry Transport latency and Quality of Service performance in remote laboratory settings, specifically for the control of a spectrometer. Building upon previous work in [20], this research specifically investigates the performance characteristics of two prominent MQTT broker implementations: AWS IoT Core and cloud-based HiveMQ.

The primary objectives of this work are the following:

- Quantify and Compare MQTT Latency: To rigorously measure and compare the end-to-end latency of MQTT communication when utilizing AWS IoT Core and HiveMQ brokers for transmitting control commands to, and receiving data from, a remote fiber optic spectrometer.
- Develop a Comprehensive QoS Benchmark: To establish a detailed QoS benchmark for both broker implementations, evaluating their performance across different MQTT QoS levels (0, 1, and 2) in the context of remote spectrometer operations.
- Characterize Latency Profiles: To generate a comprehensive latency profile for each broker, identifying factors that influence communication delays and assessing the impact of network conditions on overall system responsiveness.
- Enhance Understanding of Remote-Control Performance: To provide critical insights into how the choice of MQTT broker and its configuration impacts the responsiveness, reliability, and precision of remote spectrometer control.
- Contribute to Reliable Remote Laboratory Systems: To contribute to the development of more reliable, low-latency, and efficient remote laboratory frameworks, thereby enhancing the accessibility and effectiveness of advanced scientific experimentation.

IV. HIVEMQ CONFIGURATION AND SECURITY

When leveraging HiveMQ Cloud for remote spectrometer control, many of the underlying infrastructure components and network configurations are managed by HiveMQ, simplifying deployment. However, robust security and appropriate client configurations remain paramount to ensure the integrity, confidentiality, and availability of experimental data and control commands. The inherent scalability and reliability of HiveMQ's architecture, including its Quality of Service levels, provide a strong foundation for dependable data exchanges in our context.

HiveMQ Cloud instances provide specific hostnames and ports for MQTT connectivity. Our connections to HiveMQ Cloud utilize Transport Layer Security to encrypt data in transit, safeguarding sensitive measurements and control signals from eavesdropping and tampering. HiveMQ Cloud automatically provisions TLS, meaning our client will connect to the secure MQTT port 8883.

Authentication verifies the identity of any device or application attempting to connect to our HiveMQ Cloud broker. For the remote spectrometer experiment, this involves authenticating both the Latte Panda SBC and the remote user dashboard. Robust authentication prevents unauthorized devices from connecting to and potentially disrupting the experiment.

To authenticate we generate a unique client certificate and private key for your Latte Panda SBC. These are then registered with HiveMQ Cloud. During connection, both the client and the broker present their certificates to each other, establishing mutual trust. This ensures that only your authorized spectrometer device can connect.

Authorization determines what an authenticated client is permitted to do, such as publish data to specific topics or subscribe to control commands. Implementing granular Access Control Lists is vital to ensure that devices and users only have access to the topics necessary for their function, following the principle of least privilege.

We have designed a clear topic hierarchy for your spectrometer experiment:

- spectrometer/{device_id}/control/rgb_led: For controlling the LED.
- spectrometer/{device_id}/control/acquisition_start: For initiating spectral acquisition.
- spectrometer/{device_id}/data/spectrum: For publishing spectral data.
- spectrometer/{device_id}/data/status: For publishing device status.

Within the HiveMQ Cloud console, we have defined rules to grant/deny PUBLISH and SUBSCRIBE permissions for specific clients (identified by their client ID, username, or certificate details) on specific topics.

Latte Panda SBC: SUBSCRIBE access to spectrometer/{device_id}/control/# topics and PUBLISH access to spectrometer/{device_id}/data/# topics.

Remote Dashboard: PUBLISH access to spectrometer/{device_id}/control/# topics and SUBSCRIBE access to spectrometer/{device_id}/data/# topics.

No client has broader access than absolutely required. For example, the spectrometer client should not be able to publish to control topics, and the dashboard should not be able to subscribe to internal device configuration topics.

Our Python script on the Latte Panda SBC uses a paho-mqtt client library to connect to HiveMQ Cloud. The configuration includes the secure broker address, port, and authentication credentials.

By meticulously configuring these aspects within HiveMQ Cloud and on our Latte Panda SBC, we can establish a secure, reliable, and efficient communication channel for our remote spectrometer experiment.

V. METHODOLOGY FOR LATENCY AND QUALITY OF SERVICE MEASUREMENT

This section outlines the systematic methodology employed to quantify and compare the end-to-end latency and assess the Quality of Service performance of MQTT communication within the remote spectrometer control framework. The objective is to provide a robust experimental procedure for benchmarking both AWS IoT Core and HiveMQ Cloud brokers, building on established practices for IoT protocol performance evaluation.

In the context of this experiment, end-to-end latency is defined as the total time elapsed for a complete operational cycle, encompassing both control command execution and data feedback. First, we evaluate control path latency - the duration from the initiation of a control command (e.g., LED color change, spectral acquisition trigger) from the web dashboard until the physical actuation of the device (e.g., LED illumination change) on the Latte Panda SBC. Then we'‘ll assess data feedback path latency - the duration from the moment the StellarNet spectrometer acquires spectral data until that processed data is accurately displayed on the remote web dashboard.

The latency measurements are repeated for each of the MQTT Quality of Service levels, as QoS significantly impacts delivery guarantees and performance [9, 16]. QoS 0 messages are sent without acknowledgment, typically offering the lowest latency but providing no delivery guarantees, which can potentially lead to message loss. In contrast, QoS 1 messages are guaranteed to arrive, though duplicates may occur, and this involves an acknowledgment handshake that can potentially increase latency compared to QoS 0. For the strongest delivery guarantee, QoS 2 ensures messages arrive exactly once; however, this involves a more complex four-way handshake, resulting in the highest latency [16].

The impact of each QoS level on average latency, message loss rate, and jitter will be systematically evaluated for both control and data paths. Studies indicate that QoS settings significantly influence middleware performance [6].

VI. RESULTS AND DISCUSSION

This section will present and analyze the quantitative results obtained from the latency and Quality of Service benchmarking experiments. It will delve into the observed performance characteristics of MQTT communication under various configurations, with a particular focus on the impact of QoS levels and a comparative analysis between AWS IoT Core and HiveMQ Cloud broker implementations.

A. LATENCY FROM QOS DEPENDENCY

The experimental data reveal the relationship between MQTT QoS levels and observed end-to-end latency for both control command and data feedback paths. Varying QoS settings significantly impact message delivery times and reliability, reflecting the trade-offs inherent in the MQTT protocol's design.

Fig. 1 visually represents the latency characteristics in milliseconds for MQTT Quality of Service levels 0, 1, and 2, which is critical for understanding performance in remote spectrometer control.

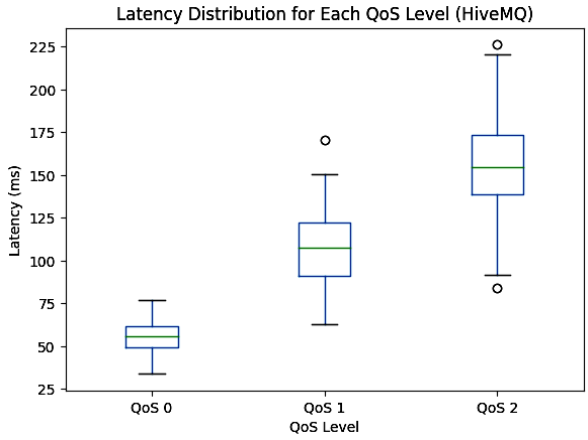

Fig. 1. Latency distribution per QoS level

For QoS 0 Latency Distribution, the median latency is approximately 50 ms, with a very narrow interquartile range of about 10 ms (from 45 ms to 55 ms). This indicates exceptional consistency and the lowest latency, characteristic of its "fire and forget" delivery model. An outlier at 30 ms shows rare, even faster instances.

QoS 1 Latency Distribution shows a notable increase, with a median latency of approximately 100 ms. Its interquartile range widens to about 20 ms (from 90 ms to 110 ms), signifying greater variability. An outlier at 155 ms suggests occasional higher delays. This increased latency and variability are attributed to the "at least once" delivery guarantee's acknowledgment handshake.

Finally, QoS 2 Latency Distribution exhibits the highest median latency, around 155 ms. The interquartile range is significantly broader, approximately 45 ms (from 130 ms to 175 ms), demonstrating the greatest variability. Multiple outliers, including a substantial one at 265 ms, underscore extreme delays. This reflects the complex four-way handshake required for "exactly once" delivery, resulting in the highest overhead and least predictable timing.

In Overall Trends, the plot clearly illustrates a monotonic increase in both median latency and latency variability (jitter) as the QoS level rises from 0 to 2. This confirms the fundamental MQTT trade-off: higher reliability comes at the cost of increased delay and reduced consistency. For remote spectrometer control, where "low latency is crucial for responsive and precise operation" and "reliable message delivery is equally vital", this analysis highlights the challenge of balancing these competing demands. While QoS 0 offers speed and consistency, higher QoS levels, though guaranteeing delivery, introduce significant jitter and delays, which can impact the "responsive and precise operation" necessary for effective experimental execution.

B. COMPARISON OF AWS IOT CORE AND HIVEMQ CLOUD PERFORMANCE

This subsection will present a detailed comparative analysis of the performance of AWS IoT Core and HiveMQ Cloud brokers across all measured metrics, leveraging statistical methods and percentile distributions to provide a nuanced understanding of their respective strengths and weaknesses in the context of remote spectrometer control.

Fig. 2 compares the latency distribution between AWS IoT Core and HiveMQ brokers across MQTT QoS levels 0, 1, and 2. This visualization offers crucial insights into consistency and tail-end performance, essential for understanding the "impact of network conditions on overall system responsiveness" in remote spectrometer control.

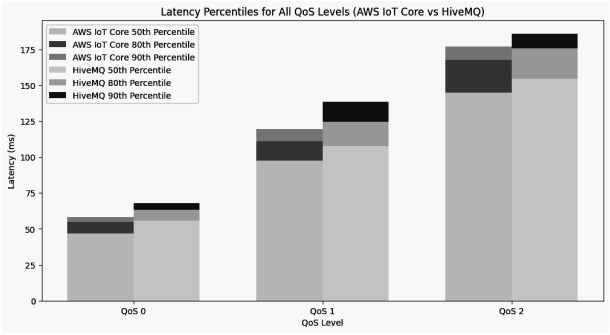

Fig. 2. Latency percentiles comparison

For QoS 0, AWS IoT Core demonstrates superior performance. Its 50th percentile latency is 48 ms, with the 80th at 52 ms and 90th at 58 ms. HiveMQ exhibits higher latencies: 50th percentile at 56 ms, 80th at 60 ms, and 90th at 68 ms. This indicates AWS IoT Core has lower typical and tighter worst-case latencies for QoS 0.

Moving to QoS 1, both brokers show an expected latency increase due to the acknowledgment handshake. AWS IoT Core maintains its advantage, with a 50th percentile of 98 ms, 80th of 110 ms, and 90th of 120 ms. HiveMQ's QoS 1 performance is slightly higher, with a 50th percentile of 108 ms, 80th of 125 ms, and 90th of

138 ms. The wider spread between percentiles for both brokers compared to QoS 0 indicates increased variability, but AWS IoT Core still shows a more compact distribution.

For QoS 2, providing "exactly once" delivery, latencies are significantly higher and more varied. AWS IoT Core records a 50th percentile latency of 143 ms, 80th of 165 ms, and 90th of 178 ms. HiveMQ's latencies are higher, with a 50th percentile of 155 ms, 80th of 175 ms, and 90th of 185 ms. The substantial increase in 90th percentile values for both brokers at QoS 2 underscores the impact of demanding delivery guarantees on tail latency.

Overall, AWS IoT Core consistently outperforms HiveMQ across all QoS levels and percentiles, suggesting a more responsive communication channel for latency-sensitive applications. Secondly, for both brokers, latency monotonically increases across all percentiles as the QoS level rises from 0 to 2, reinforcing MQTT's inherent trade-off between enhanced delivery guarantees and increased communication delays.

Furthermore, the chart effectively illustrates latency jitter and tail latency. The widening gap between the 50th and 90th percentiles as QoS increases, particularly for QoS 2, indicates a significant rise in latency variability. This is especially pronounced for HiveMQ, which consistently shows a greater spread compared to AWS IoT Core, implying less predictable message delivery times. The "Latency Distribution Analysis" emphasizes the importance of understanding these worst-case scenarios, as high 90th percentile values mean 10% of messages experience significantly longer delays, critical for time-sensitive control commands. While "reliable message delivery" is vital, increased jitter and tail latency at higher QoS levels, particularly for HiveMQ, challenge the "responsive and precise operation" of remote spectrometer control systems.

Fig. 3 provides a statistical visualization of the probability distribution of latency measurements for both AWS IoT Core and HiveMQ brokers across MQTT Quality of Service levels. This granular view is crucial for understanding typical latency, variability, and spread, directly impacting the "responsive and precise operation" required for remote spectrometer control.

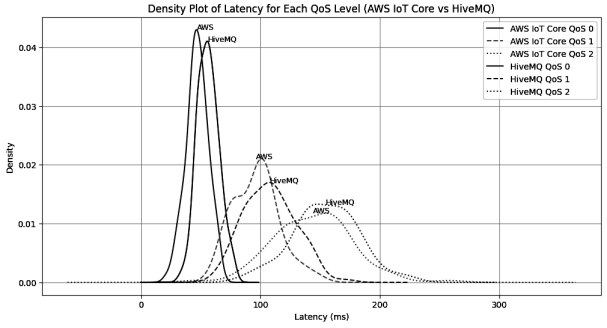

Fig. 3. Probability distribution of latency measurements

For QoS 0, which represents the "fire and forget" delivery, both brokers show sharp, distinct peaks indicating low and consistent latency. AWS IoT Core QoS 0 (solid blue line) peaks at approximately 48 ms, displaying a very narrow distribution. HiveMQ QoS 0 (solid red line) peaks slightly higher, around 56 ms, and also exhibits a very narrow, concentrated distribution. The AWS IoT Core curve is slightly to the left and taller, signifying a marginally lower modal latency and possibly even less variability compared to HiveMQ for QoS 0. This indicates that AWS IoT Core delivers messages faster and with greater consistency at this lowest reliability level.

Moving to QoS 1, which ensures "at least once" delivery, both distributions shift to higher latencies and become noticeably wider, reflecting increased variability. AWS IoT Core QoS 1 (dashed orange line) peaks around 98 ms and shows a moderately wide spread. HiveMQ QoS 1 (dashed purple line) peaks at a higher latency, approximately 108 ms, and its distribution appears slightly wider and flatter than AWS IoT Core QoS 1. The curves for QoS 1 are clearly separated from those of QoS 0, illustrating the increased overhead from the acknowledgment handshake. AWS IoT Core maintains its advantage, showing a lower modal latency and a more concentrated distribution compared to HiveMQ.

Finally, for QoS 2, offering "exactly once" delivery through a complex four-way handshake, the distributions shift further to the right, indicating the highest latencies, and become significantly broader and flatter, revealing substantial latency variability. AWS IoT Core QoS 2 (dotted green line) peaks around 143 ms, demonstrating a wide spread of latency values. HiveMQ QoS 2 (dotted brown line) peaks at an even higher latency, approximately 155 ms, and exhibits the broadest and flattest distribution among all the curves, indicating the greatest variability and least predictable timing. The extensive overlap between the QoS 2 distributions and the higher latency tails of QoS 1 highlights the increased uncertainty introduced by this highest reliability level.

Overall, the density plot provides several critical statistical insights. Firstly, it visually confirms a monotonic increase in latency for both brokers as the QoS level increases from 0 to 2. This is evident from the progressive rightward shift of the peak densities, directly illustrating the trade-off between reliability and speed inherent in MQTT. Secondly, there is a clear and consistent trend of increasing latency variability (jitter) with higher QoS levels, as indicated by the widening and flattening of the density curves. This means that not only do messages take longer on average at higher QoS, but their arrival times also become less predictable, which is a key concern for "real-time control" in remote laboratory settings.

Furthermore, AWS IoT Core consistently outperforms HiveMQ across all QoS levels. For each QoS level, AWS IoT Core's density curve peaks at a lower latency value and generally appears more concentrated (taller and narrower) than HiveMQ's corresponding curve. This suggests that AWS IoT Core offers both lower

typical latency and greater consistency in message delivery, a significant advantage for maintaining the "responsive and precise operation" of remote spectrometer control. HiveMQ, while functional, shows higher modal latencies and greater variability, particularly at QoS 2, which could translate to less predictable control and feedback within the remote laboratory environment. This detailed statistical analysis of the latency distributions underscores the importance of choosing the appropriate broker and QoS level to balance the need for "reliable message delivery" with the strict requirements for low latency and consistent performance in demanding scientific applications.

VII. CONCLUSION

The statistical analysis of latency distribution, as presented through density plots and percentile comparisons for AWS IoT Core and HiveMQ across various MQTT QoS levels, yields critical insights for remote spectrometer control.

The density plots and percentile analyses consistently demonstrate that AWS IoT Core generally exhibits lower modal (peak density) latencies and tighter distributions across all QoS levels compared to HiveMQ.

This statistical examination further underscores the inherent trade-off in the MQTT protocol: higher QoS levels, while offering stronger delivery guarantees, invariably lead to higher average latencies and increased jitter. The widening gap between lower and higher percentiles as QoS increases highlights this rise in latency variability, which is particularly pronounced in HiveMQ's performance. Such variability can significantly impact the responsive and precise operation crucial for real-time remote control.

In conclusion, the detailed statistical analysis confirms that AWS IoT Core consistently provides lower latency and greater consistency across all QoS levels, which is essential to definitively guide the selection of the optimal broker and QoS configuration that balances the critical requirements of reliability, responsiveness, and precision for remote spectrometer control.

VIII. CONFLICTS OF INTEREST

The authors declare no conflicts of interest.

IX. DECLARATION ON GENERATIVE AI

During the preparation of this work, the author(s) used ChatGPT, Grammarly in order to: Grammar and spelling check, Paraphrase and reword. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] Ammar, M., Russello, G., & Crispo, B. (2017). Internet of Things: A survey on the security of IoT frameworks. *Journal of Information Security and Applications*, *38*, 8. DOI: https://doi.org/10.1016/j.jisa.2017.11.002

[2] Dizdarević, J., Michalke, M., & Jukan, A. (2023). Engineering and Experimentally Benchmarking Open Source MQTT Broker Implementations. *arXiv (Cornell University)*. DOI: https://doi.org/10.48550/arxiv.2305.13893

[3] Handosa, M., Gračanin, D., & Elmongui, H. G. (2017). Performance evaluation of MQTT-based internet of things systems. *2018 Winter Simulation Conference (WSC)*, 4544. DOI: 10.1109/WSC.2017.8248196

[4] Hmissi, F., & Ouni, S. (2024). A survey on application layer protocols for iot networks. *arXiv preprint arXiv:2405.15901*. DOI: https://doi.org/10.48550/arxiv.2405.15901

[5] Jeddou, S., Fernández, F., Díez, L., Baïna, A., Abdallah, N., & Agüero, R. (2022). Delay and Energy Consumption of MQTT over QUIC: An Empirical Characterization Using Commercial-Off-The-Shelf Devices. *Sensors*, *22*(10), 3694. DOI: https://doi.org/10.3390/s22103694

[6] Kang, Z., Canady, R., Dubey, A., Gokhale, A., Shekhar, S., & Sedlacek, M. (2020, December). A study of publish/subscribe middleware under different iot traffic conditions. In *Proceedings of the International Workshop on Middleware and Applications for the Internet of Things* (pp. 7-12). DOI: https://doi.org/10.1145/3429881.3430109

[7] Kirwan, R. F., Abbas, F., Atmosukarto, I., Loo, A. W. Y., Lim, J. H., & Yeo, S. (2023). Scalable agritech growbox architecture. *Frontiers in the Internet of Things*, *2*, 1256163. DOI: https://doi.org/10.3389/friot.2023.1256163

[8] Mishra, B., Mishra, B., & Kertész, A. (2021). Stress-Testing MQTT Brokers: A Comparative Analysis of Performance Measurements. *Energies*, *14*(18), 5817. DOI: https://doi.org/10.3390/en14185817

[9] Palmese, F., Redondi, A. E. C., & Cesana, M. (2022). Adaptive Quality of Service Control for MQTT-SN. *Sensors*, *22*(22), 8852. DOI: https://doi.org/10.3390/s22228852

[10] Patil, V. B., & Khodade, P. S. (2025). Development and applications of an affordable DIY optical spectrometer using a webcam. *Journal of Optics*, 1-4. DOI: https://doi.org/10.1007/s12596-025-02714-7

[11] Popoola, O., Rodrigues, M., Marchang, J., Shenfield, A., Ikpehai, A., & Popoola, J. (2023). A critical literature review of security and privacy in smart home healthcare schemes adopting IoT & blockchain: Problems, challenges and solutions. *Blockchain Research and Applications*, *5*(2), 100178. DOI: https://doi.org/10.1016/j.bcra.2023.100178

[12] Rager, D., & Andersen, A. (2021, October). NREL Stratus-Enabling Workflows to Fuse Data Streams, Modeling, Simulation, and Machine Learning. In *Smoky Mountains Computational Sciences and Engineering Conference* (pp. 227-246). Cham: Springer International Publishing. URL: https://www.osti.gov/biblio/1899966

[13] Sánchez, C. B., Redondo, R. P. D., Vilas, A. F., & Sánchez, Á. (2018). Spectrophotometers for labs: A cost-efficient solution based on smartphones. *Computer Applications in Engineering Education*, *27*(2), 371. DOI: https://doi.org/10.1002/cae.22081

[14] Silva, D. M. A. da, Carvalho, L. I., Soares, J., & Sofia, R. C. (2021). A Performance Analysis of Internet of Things Networking Protocols: Evaluating MQTT, CoAP, OPC UA. *Applied Sciences*, *11*(11), 4879. DOI: https://doi.org/10.3390/app11114879

[15] Silva, T. F. da, Rodrigues, C. L., Tabacniks, M. H., Pereira, H. D. C., Saramela, T. B., & Guimarães, R. O.

(2020). Ion beam analysis and big data: How data science can support next-generation instrumentation. *Nuclear Instruments and Methods in Physics Research Section B Beam Interactions with Materials and Atoms*, *478*, 111. DOI: https://doi.org/10.1016/j.nimb.2020.05.027

[16] Tareq, R. W., & Khaleel, T. A. (2021). Implementation of MQTT Protocol in Health Care Based on IoT Systems: A Study. *International Journal of Electrical and Computer Engineering Systems*, *12*(4), 215. DOI: https://doi.org/10.32985/ijeces.12.4.5

[17] Tunēns, G., Einbergs, E., Laganovska, K., Zolotarjovs, A., Vilks, K., Skuja, L., & Šmits, K. (2024). Optical fiber-based open source low cost portable spectrometer system. *HardwareX*, *18*. DOI: https://doi.org/10.1016/j.ohx.2024.e00530

[18] Ullah, M., Nardelli, P. H. J., Wolff, A., & Smolander, K. (2020). Twenty-One Key Factors to Choose an IoT Platform: Theoretical Framework and Its Applications. *IEEE Internet of Things Journal*, *7*(10), 10111. DOI: 10.1109/JIOT.2020.3000056

[19] Yao, C., Chen, M., Yan, T., Ming, L., Cheng, Q., & Penty, R. (2023). Broadband picometer-scale resolution on-chip spectrometer with reconfigurable photonics. *Light: Science & Applications*, *12*(1), 156. DOI: https://doi.org/10.1038/s41377-023-01195-2

[20] Krupych, A., Karbovnyk, I. (2025). Computerized Optical Experiments with Portable Fiber Optic Spectrometers and Amazon Web Services Cloud Integration. *Electronics and information technologies, 29,* 123-134. DOI: http://dx.doi.org/10.30970/eli.29.11

Andriy Krupych was born in Lviv, Ukraine, on May 29, 1991. He received the B.S. (2012) and M.S. (2013) degrees in computer science from the Faculty of Applied Mathematics and Informatics, Ivan Franko National University of Lviv. Currently PhD candidate at the Department of Radiophysics and Computer Technologies. His research interests include optics, cloud and Internet of Things (IoT).

Edgars Elsts is a PhD in Physics, Senior scientist at the Institute of Solid-State Physics, University of Latvia, one of the world's leading experts in the field of solid-state radiation physics, sensor materials for cyber-physical systems.