

Article

A Reliable Publish–Subscribe Mechanism for Internet of Things-Enabled Smart Greenhouses

Behnaz Motamedi *  and Balázs Villányi

Department of Electronics Technology, Budapest University of Technology and Economics (BME),
1111 Budapest, Hungary; villanyi.balazs@vik.bme.hu

* Correspondence: behnazmotamedi@edu.bme.hu

Abstract: Messaging protocols for the Internet of Things (IoT) play a crucial role in facilitating efficient product creation and waste reduction, and in enhancing agricultural process efficiency within the realm of smart greenhouses. Publish–subscribe (pub-sub) systems improve communication between IoT devices and cloud platforms. Nevertheless, IoT technology is required to effectively handle a considerable volume of subscriptions or topic adjustments from several clients concurrently. In addition, subscription throughput is an essential factor of the pub-sub mechanism, as it directly influences the speed at which messages may be sent to subscribers. The primary focus of this paper pertains to a performance assessment of the proposed message categorization architecture for the Message Queue Telemetry Transport (MQTT) broker. This architecture aims to establish a standardized approach to pub-sub topics and generate new topics with various performance characteristics. We also standardize the form of MQTT protocol broker topic categorization and payload based on greenhouse specifications. The establishment of topic classification enhances the operational effectiveness of the broker, reduces data volume, and concurrently augments the number of messages and events transmitted from the greenhouse environment to the central server on a per-second basis. Our proposed architecture is validated across multiple MQTT brokers, including Mosquitto, ActiveMQ, Bevywise, and EMQ X, showing enhanced flexibility, extensibility, and simplicity while maintaining full compatibility with greenhouse environments. Key findings demonstrate significant improvements in performance metrics. The message processing time for the proposed Active MQ broker was increased approximately five-fold across all QoS levels compared to the original. Subscription throughput for the Bevywise MQTT Route 2.0 broker at QoS0 reached 1453.053, compared to 290.610 for the original broker. The number of messages in the Active MQ broker at QoS0 surged from 394.79 to 1973.95. These improvements demonstrate the architecture’s potential for broader IoT applications in pub-sub systems.

Keywords: MQTT protocol broker; Internet of Things; publish–subscribe systems; data volume; subscription throughput



Citation: Motamedi, B.; Villányi, B. A Reliable Publish–Subscribe Mechanism for Internet of Things-Enabled Smart Greenhouses. *Appl. Sci.* **2024**, *14*, 6407. <https://doi.org/10.3390/app14156407>

Academic Editors: Andrew W. H. Ip, Chunsheng Zhu and Weidong Fang

Received: 1 July 2024

Revised: 17 July 2024

Accepted: 20 July 2024

Published: 23 July 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The Internet of Things (IoT)-enabled smart greenhouse is an innovative agricultural system that employs cutting-edge technology to improve crop yield [1–4]. These systems are a technologically sophisticated variant of conventional conservatories designed to improve agricultural productivity, efficiency, and sustainability. The continuous collection of sensory data from sensors, such as temperature, humidity, light intensity, soil moisture, and other variables, is required to monitor and control environmental variables, considerably enhance the greenhouse’s performance, and transmit environmental information to cloud-based systems. Remaining up-to-date with greenhouse climate characteristics and modifying them to correspond with the requirements of various crops has become a formidable obstacle [5–7]. Therefore, it is crucial to establish standardized seamless connections between the cloud infrastructure and IoT systems [8,9]. Similarly, the articles discuss many

machine-to-machine (M2M) protocols employed for the transmission of IoT application information. These technologies include Constrained Application Protocol (CoAP) and Message Queue Telemetry Transport (MQTT) [4,10,11]. CoAP is a RESTful platform that employs a request/response structure. It is evolved from the Hypertext Transfer Protocol (HTTP) and developed specifically to adjust to the limitations of restricted systems [12,13]. Nevertheless, in the context of cloud connectivity, a crucial aspect of devices is their ability to autonomously transfer data without being prompted for an additional request. Hence, the MQTT publish–subscribe (pub-sub) system presents the capability to transmit data to the cloud regularly [10,14–17], without necessitating the reception of a request. In addition, pub-sub structures establish connections between terminals or clients, such as IoT products and cloud platforms, through the assistance of a broker [18]. Pub-sub systems guarantee the transmission of information to the server by employing Transmission Control Protocol (TCP) as the communication protocol [19]. These systems supply two essential components for the messages: the topic, which is an organized list that specifies the context to which the sent data pertain, and the payload [4]. In terms of the topic classification between pub-sub clients and the broker, the establishment of a hierarchical categorization mechanism for messages facilitates the optimization of message routing and processing on the broker’s end. Hence, the unorganized nature of the data presents an appropriate level of adaptability. However, it is important to acknowledge that this flexibility comes with certain limitations. For instance, the size of the topic and payload may range from tens to hundreds of bytes, as demonstrated in the studies conducted by researchers [15,20–22]. Additionally, the implementation of a standardized pub-sub structure within the context of MQTT presents a significant challenge, as illustrated by the investigation performed by the authors [4,10,23,24]. The deployment of pub-sub mechanisms through MQTT brokers may be categorized into two distinct types, namely, scalable and non-scalable, depending on the particular requirements of IoT applications. In circumstances where there is an imperative to provide support to a substantial quantity of publishers, subscribers, and messages, the presence of a scalable MQTT broker becomes crucial. In cases where scalability is not a fundamental desire, a non-scalable broker may be adequate for limited installations [25].

To address these challenges, this research paper proposes a Smart Greenhouse Protocol (SGP) based on the MQTT protocol. The principal objectives of this research are as follows:

- Standardize the topic and pub-sub system to minimize server processing.
- Reduce complexity and cost while providing optimal extensibility and flexibility to meet the demands of greenhouse systems.
- Demonstrate the efficacy of the proposed strategy even in challenging conditions, enabling reliable information transmission during significant events.
- Compare the number of transferred messages between traditional and proposed architectures in a greenhouse environment.
- Conduct an extensive evaluation of topic categorization and pub-sub systems (scalable and non-scalable MQTT) regarding message processing time and subscription throughput.

This research discusses four MQTT brokers as low-resource-consuming messaging applications: Mosquitto 1.4.15, Bevywise MQTT Route 2.0 (known for scalability), ActiveMQ 5.15.8, and EMQ X 4.0.8 (known for non-scalability). These brokers are evaluated based on their suitability for the pub-sub communication framework, considering the reliability specifications of the client’s pub-sub structure.

The diagram depicted in Figure 1 demonstrates the three-tier architecture adopted for messaging communication in an IoT greenhouse. The paper is organized as follows. Section 2 discusses related works on the existing solutions of various messaging protocols. Section 3 is about the research methodology, in which Section 3.1 describes our proposed protocol architecture. Then, we introduce in detail the overall system of our proposed message standardization and topic configuration in Section 3.2. In Section 3.3, the pub-sub system for MQTT is proposed. Sections 4 and 5 discuss the performance evaluation, summarize, and conclude the paper.

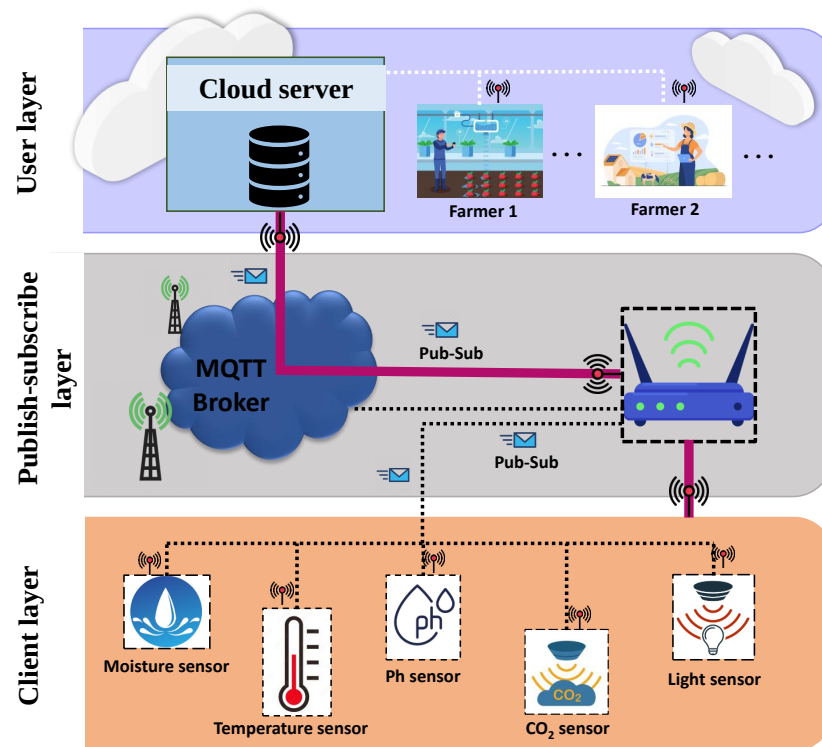


Figure 1. MQTT protocol publish–subscribe message communication between sensors and the IoT-based greenhouse environment.

2. Related Approaches

The establishment of the messaging protocol in a smart greenhouse application is contingent upon the unique circumstances of the system, including factors such as connection speed, limited bandwidth, and device capabilities. Selecting a protocol that is both efficient and scalable is imperative in achieving the objectives of the smart greenhouse system while ensuring that the necessary functionality is delivered.

The work of reference [26] explored a comprehensive academic investigation of IoT messaging protocols from the standpoint of information transmission for each message and evaluated their performance in a research environment. The findings discussed in the study indicate that MQTT represents the appropriate protocol for data extraction, citing its efficiency, scalability, and reliability. In comparison, CoAP, while excelling in transmission time due to its UDP-based protocol, lacks the reliability characteristic of MQTT, which is crucial for applications requiring assured message delivery.

In 2020, reference [16] describes the significance of IoT messaging protocols in smart farming along with the challenges involved in determining the appropriate protocol for a particular application. The research paper subsequently provides an overview of various IoT communications protocols, including MQTT, CoAP, XMPP, etc., and examines their efficacy based on major factors such as latency, energy and bandwidth specifications, throughput, reliability, and security. According to the author’s analysis, MQTT represents one of the majority of widespread and trustworthy protocols for smart farming programs owing to its accessibility, dependability, and industry-wide usage, whereas CoAP and XMPP technologies exhibit certain drawbacks, including an enormous network packet loss rate (PLR), message loss, and excessive latency. Therefore, this study aims to develop a new methodology within the MQTT protocol to enable the supervision and surveillance of an advanced greenhouse.

The study executed by [27] presents an approach to employ a key exchange technique as a lightweight and reliable authentication mechanism between an interface and a smart device. This investigation incorporates the MQTT communication protocol to execute the

monitoring and control functionalities. Nevertheless, the implementation of the platform using traditional MQTT protocol has specific constraints, such as categorizing each device as a distinct node, which increases the cost and complicates the architecture. When dealing with big data, it is advisable to avoid limiting the exploration to just three separate subjects for categorizing the given data. As a response, a revised MQTT protocol structure suggest to better cater to the specific requirements of greenhouse settings, optimizing both performance and management overhead.

The present work examines the efficacy of five distributed MQTT brokers utilizing the recommended BORDER framework architecture, as described in [28]. Additionally, significant metrics such as latency, throughput, and scalability are investigated during the testing and analysis phase of pub-sub mechanisms in this study. Nonetheless, it is worth noting that data transmission between the brokers can result in greater overhead compared with employing a single broker. When numerous clients of the same broker are interested in the same resource, each client develops a distinct connection with the broker supplying the resource. Consequently, this may culminate in insufficient network capacity utilization [29]. Subsequently, we present a novel architecture with a singular broker that not only eliminates complexity but also improves scalability and reliability.

Similarly, Ref.[30] concentrates on an improved MQTT technology for edge processing with a double broker in the fog and cloud, including an analytical framework for calculating performance indicators of the proposed architecture. The authors must contemplate a software-defined multi-tier processing system for low latency in IoT and compare its efficacy to the standard MQTT-based IoT approach. However, it should be noted that a comprehensive description of the modified protocol's pub-sub system has not been provided, and each topic is assigned to a unique broker, resulting in more convoluted message navigation, greater latency, and delayed message delivery. In the meantime, this can be particularly challenging for real-time scenarios that require rapid and trustworthy communications [31]. Alternately, the current study proposes a novel pub-sub mechanism for transmitting data from the client to a single broker capable of increasing the pace at which messages are transmitted to the subscribers.

Another examination [32] conducted a study to discover the most productive network configuration as well as enhance the total number of interactions while minimizing resource waste. In addition, the primary objective of this research is to provide an integer linear algorithm for improving the network configuration of a traditional MQTT broker. A considerable quantity of nodes is taken into account in the framework of this paper, which may result in eliminated or delayed messages because the conventional MQTT supports a maximum packet size of 256 MB, which may be inefficient as the total quantity of devices expands [33]. We establish a distinct MQTT topic structure, along with a single node for commands and a structure for message payloads, to enhance the performance and efficacy of MQTT commands.

In [34], researchers describe the software-defined multi-tier edge processing approach by enhancing the standard MQTT system for edge estimations to contain five external brokers in the fog node, an extensive number of nodes, and a single central broker in the cloud. Likewise, the developed framework evaluates latency, transmission delay, processing delay, and the total number of packets transmitted to the cloud. Nonetheless, as the number of nodes in an architecture progresses, so does the quantity of data and network traffic connected with the broker. Additionally, the distribution of data from the client to the server becomes extremely complex. By presenting the proposed architecture of the MQTT protocol topic classification and pub-sub system, we aim to minimize the amount of data and server processing time while simultaneously enhancing the scalability, reliability, and flexibility of the protocol. The implication of reliability focuses on Quality of Service (QoS) regarding both the quality of information acquired and accurate decision-making, and scalability implies the ability to accommodate a growing variety of clients [35].

The study by [4] intended to evaluate the reliability of a message structure that streamlines the pub-sub topic and payload adopted by IoT products and the cloud service

provider. This study assessed a message framework for pub-sub systems that are energy-efficient, message-aware, and end-to-end secure. Although the categorization of topics within the MQTT protocol framework was found to be insufficient and inadequate in this study, it did not have an influence on the greenhouse environment with extensive data. To elaborate, a data capacity of five bytes is deemed inadequate for a comprehensive and versatile framework. Moreover, the research on expandability and flexibility has not been accomplished, despite being one of the primary objectives of our proposed methodology.

In summary, MQTT emerges as a protocol for smart greenhouse applications due to its extensive adoption across IoT ecosystems, proven efficiency, reliability, and scalability. While alternatives like CoAP and XMPP offer specific advantages, such as lightweight communication and versatile messaging formats, they may lack the robustness and industry support required for extensive IoT deployments in environments demanding consistent data delivery and management. The principles of event-driven architecture and distributed systems are the foundation of the research on the topic categorization within the MQTT framework and the evaluation of interactions between various terminals (including broker, publisher, and subscriber) through the pub-sub system. The arrangement of distributed data is made simpler by the topic categorization in MQTT, which enables receivers to select and acquire only relevant data. At present, there is a restricted number of scholarly articles that delve into the topic of pub-sub systems and analyze the performance of MQTT through topic classification. Our proposed research aims to develop an architectural framework that specifically incorporates three essential characteristics—flexibility, extensibility, and simplicity—that make it unique to the greenhouse environment. In addition, a new method is suggested to create a consistent communication protocol in MQTT employing a unique structural approach. This method entails dividing data into several subjects and assigning priority to a particular issue in cases of crucial or urgent conditions in the greenhouse environment. The objective of the proposed research is to improve the usability of MQTT in smart greenhouse environments by refining its protocol architecture and optimizing pub-sub mechanisms to satisfy the changing requirements of agricultural IoT systems.

3. Research Methodology

This section introduces our reliable SGP, which incorporates an optimized MQTT broker. The proposed protocol SGP enables the transmission of structured messages with various topics, presenting flexible and extensible capabilities for connecting to a cloud platform through pub-sub systems. We also highlight the importance of reducing complexity and cost, minimizing processing time, and decreasing the number of transferring messages for both the MQTT broker framework and the primary server.

3.1. Architecture and Protocol Description

In this section, we propose an architecture and protocol for the remote monitoring of greenhouse environmental parameters and control systems to optimize plant growth and resource utilization. Our advanced architectural framework enables the remote monitoring of greenhouse environmental variables and precise management of critical systems. This framework, as illustrated in Figure 2, is composed of four distinct levels, which together function as an interconnected system.

The first layer consists of strategically positioned sensors inside the greenhouse that are intended to collect essential environmental variables, such as temperature, humidity, the condition of the actuator, and the operational state of all accessible equipment. Following this, the gathered data and subsequent greenhouse events are transmitted to the SGP MQTT protocol through the environmental parameter collection device situated inside the greenhouse environment.

As depicted in layer 2 of Figure 2, our proposed solution introduces an innovative approach by emphasizing the standardizing the transmission of messages from the greenhouse to the primary server. This aspect has been disregarded or inadequately addressed in several other articles, leading to the need for more processing [36]. Therefore, this increases

the complexity of the intelligent system, as explained in Section 2. The subscriber is enrolled to get updates on specific topics, which will be defined, while the publisher distributes content on the indicated themes. Both entities facilitate the transmission of data gathered from diverse sensors to access the Sub-Pub systems of the MQTT broker. Subsequently, the data are transmitted hierarchically to the primary server, as seen in layer 3 of Figure 2.

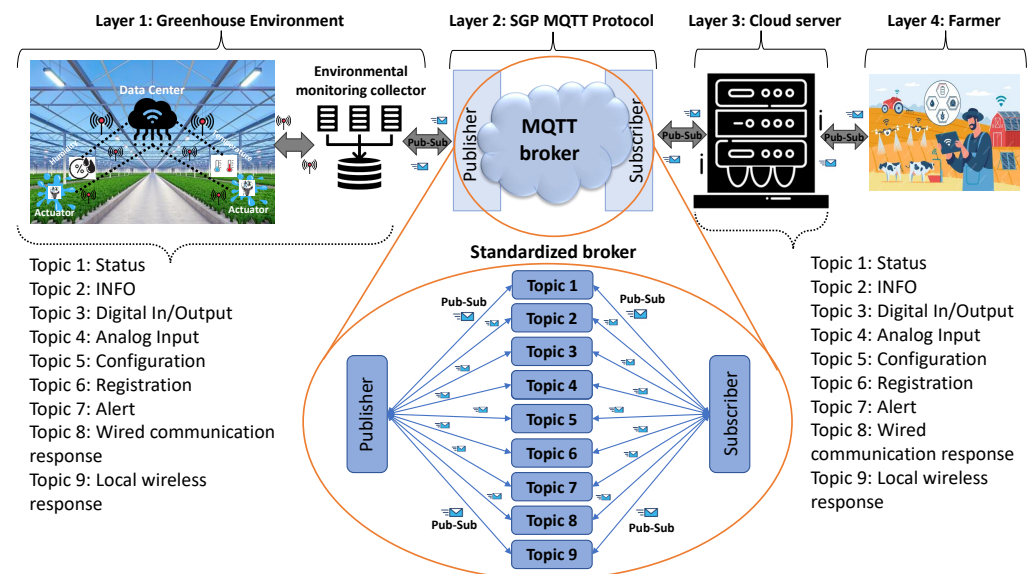


Figure 2. The proposed standardization structure of the smart greenhouse protocol for the MQTT broker using topic categorization and the publish–subscribe mechanism.

Fundamentally, all data or events inside the greenhouse are categorized and delivered to the corresponding topic on the central server using the proposed protocol. By employing this systematic approach, the farmer can consistently and immediately monitor the conditions of the greenhouse in real time. Technology enhances the cultivator’s capabilities by equipping them with crucial knowledge, thus enabling them to make well-informed and prompt decisions. Consequently, this technique optimizes plant development and enhances resource utilization inside the greenhouse. To ensure the effectiveness of this architecture, the structured MQTT broker plays a crucial role in managing the data flow and communication between different layers.

Improvements and Advantages over Traditional MQTT

The proposed MQTT broker employs a hierarchical categorization mechanism for messages, which enhances the routing and processing of messages on the broker’s end. This system entails the specification of topics and payloads, in which the subscriber receives updates on specific topics, and the publisher disseminates information that way. Our approach implements a uniform communication protocol that decreases the intricacy and processing duration on the server end. The categorization system optimizes the routing of messages to their intended recipients, reducing the quantity of extraneous data and improving the accuracy of the primary server’s processing.

The implementation of our standardized protocol and structured MQTT broker yields several significant advantages over the traditional MQTT protocol:

- **Standardized Protocol for Resource Optimization:** The protocol optimizes resource utilization by eliminating redundant processing stages.
- **High Responsiveness:** The system ensures high responsiveness, especially essential for real-time monitoring, by reducing the processing burden on the server.
- **Scalability:** The protocol facilitates scalability by effectively managing an array of publishers, subscribers, and messages.

- **Reduced Complexity:** The structured approach simplifies the overall system, making it easier to manage and maintain.
- **Cost Efficiency:** The system becomes more cost-effective by reducing the number of messages and processing time.
- **Enhanced Accuracy:** The categorization system ensures that only relevant data are transmitted, improving the accuracy of the server's processing.

In addition to improving the smart greenhouse system's general efficiency, this method makes sure it can handle any modifications or expansions that come its way. Numerous researchers have utilized IoT-enabled smart greenhouse systems to address the need for greenhouse crop growth optimization. For instance, ref. [36] evaluated lightweight cryptographic technologies between publishers and the original MQTT broker, and even between the broker and its subscribers, within the IoT network for smart irrigation systems in agriculture. Consequently, the evaluated protocol decreased resource consumption regarding memory application and execution time. However, a structured broker is not intended to be employed with the MQTT protocol to minimize the additional processing caused by publishers and brokers. Likewise, this results in a boost within irrelevant data, which diminishes the processing precision of the primary server.

Similarly, another study [37] evaluated a low-cost IoT monitoring system for appropriate irrigation and pest control operations for olive plantations situated in the eastern coastal regions of Crete, Greece, that encounter water accessibility and pest infestation issues. In addition, the proposed solution, which utilizes the original MQTT protocol, transfers beneficial information from sensors, including soil moisture, soil temperature, and weather data, to the user, thus broadening the agricultural industry's awareness of optimal irrigation management issues. Nevertheless, the lack of a well-defined topic categorization system for MQTT broker poses challenges to efficiently routing messages to their intended subscribers [38]. This situation may require further computational steps to ascertain the proper destination for each message. Thus, this may result in heightened server processing duration and potentially impact the system's responsiveness.

3.2. Message Standardization

In this subsection, our overall structure of message standardization in the proposed protocol SGP MQTT is presented, as seen in the magnified portion labeled "standardized broker" in Figure 2. The described design involves a publisher node that is directly linked to the MQTT broker. The MQTT broker is responsible for managing many topics, each of which is devoted to specific data. Based on the distinct data types and their respective applications and functionalities, each undergoes various topics between the publisher and subscriber components:

- **Status:** The client sends a CONNECT message to the broker to establish a connection. A CONNACK message and a status code are delivered by the broker. Once the connection is established, the broker maintains it until the client sends a disconnect message or the connection is terminated. Therefore, the STATUS topic notifies the server about the device's connectivity status in the greenhouse.
- **Info:** This topic permits the server to be continually kept informed of all modifications in the greenhouse.
- **Digital Output:** This topic obtains all commands for managing the entire components inside the greenhouse, including controlling various sorts of pump devices, fans, and actuators, and even turning the devices on and off. As a result, the greenhouse's controller devices can make a decision relying on this topic. For example, if the controller device is commanded to turn off the heating via this topic, it does so.
- **Digital Input:** As feedback from the tools in the greenhouse. Take turning on the heater as an example, whose command is conveyed to the digital output; the command is performed in the first action, and we only see the message of turning on, even if the heater is burned in the greenhouse. It is worth noting that receiving feedback after executing the order to switch on the heater is critical because it demonstrates the

correct execution of the command, allowing us to ensure that the heater is in good working condition, as shown in Figure 3.

- **Analog Input:** This topic can load data from the system's analog sensors, which may also declare the sensing level of the analog sensor by upgrading the analog sensor's capabilities.
- **Configuration:** When the device in the greenhouse is upgraded or the data sending time is modified, the server will be notified through this topic.
- **Registration:** This topic stores procedures for establishing and adding extra gateways to the broker, as well as withdrawing gateways from the broker.
- **Alert:** Since agricultural products are of particular importance in human consumption, and industry, considering the worst-case situation plays a critical role in preserving crops, and this topic has the task of informing the server in emergencies of the greenhouse. For instance, whenever a device becomes malfunctioning, if there is no Alert topic, the server is not notified of this circumstance.
- **Wired communication response:** Since some sensors support wired protocols such as Modbus and RS485, this topic receives and transmits information from these protocols to the main server or broker.
- **Local wireless response:** Since wireless sensors have found the most use such as ZigBee and Wi-Fi, the data related to the sensors mentioned in this topic are received and transmitted by this topic, which shows a wide range, including the CO₂, humidity, soil moisture, and temperature sensors, and so on.

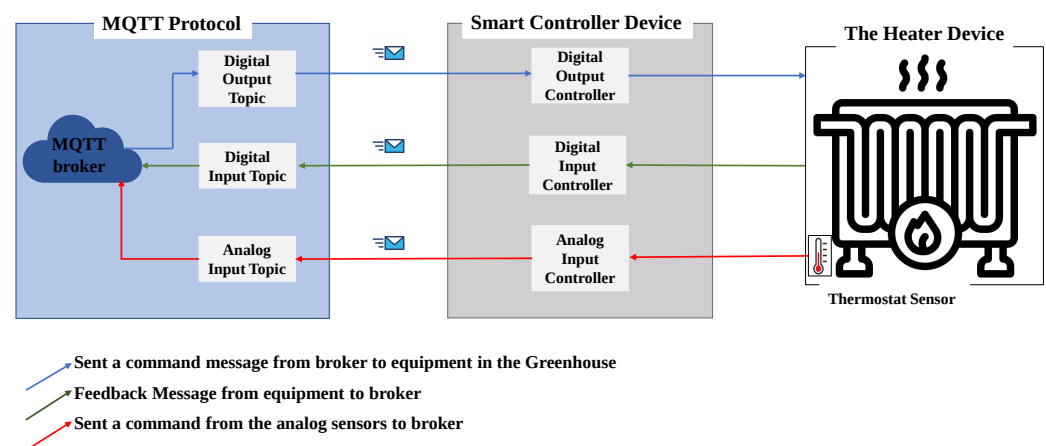


Figure 3. Digital Input/Output and Analog Input topic behavior.

The proposed MQTT broker employs a hierarchical categorization mechanism for messages, which enhances the routing and processing of messages on both the broker and server sides. This system entails the specification of topics and payloads, where the subscriber receives updates on specific topics, and the publisher disseminates information accordingly. Our approach implements a uniform communication protocol that decreases the intricacy and processing duration on the server end. The categorization system optimizes the routing of messages to their intended recipients, reducing the quantity of extraneous data and improving the accuracy of the primary server's processing. The objective of implementing a classified structure in a broker is to ensure that information and events are organized and directed to their respective topics on both the broker and server sides. This hierarchical arrangement of topics facilitates the functioning of the smart greenhouse system, making it more streamlined. Additionally, external users can conveniently access the broker using personal computers or mobile phones, allowing them to effectively manage and control categorized data through the broker. Hence, every gateway and broker functions as an IoT hub, facilitating connectivity to the central server.

3.2.1. Handling Variations in Network Conditions

Regarding the number of linked devices, the proposed approach allows for the inclusion, elimination, or substitution of devices and sensors within the greenhouse network using the Registration topic. This feature is very efficient in maintenance situations, guaranteeing that the system stays adaptable and sensitive to operational modifications.

The architecture is supposed to handle any interruptions in network connection proactively. The Alert topic is utilized for constructing local thresholds and panic messages, such as “set temp = x” or “panic = y” published before anticipated network fluctuations occur. This proactive strategy allows the system to independently regulate greenhouse climate conditions within defined thresholds, mitigating the impact of network instability on critical operations.

In summary, this architectural framework facilitates scalability, dependability, and security inside IoT systems. The distinct layers are purposefully built to address certain functions and can be readily expanded or adjusted as required. Moreover, the division of responsibilities among the many levels facilitates the maintenance and updating of the system, therefore guaranteeing its currency and security.

3.2.2. Scalability Considerations

The proposed system incorporates several mechanisms to ensure scalability in handling an increasing number of subscribers and messages:

- **Hierarchical Categorization Mechanism:** The proposed MQTT broker employs a hierarchical categorization mechanism for messages, which enhances the routing and processing of messages on both the broker and server sides. This system entails the specification of topics and payloads, where the subscriber receives updates on specific topics, and the publisher disseminates information accordingly. Our approach implements a uniform communication protocol that decreases the intricacy and processing duration on the server end. The categorization system optimizes the routing of messages to their intended recipients, reducing the quantity of extraneous data and improving the accuracy of the primary server’s processing.
- **Dynamic Topic Management:** The system can dynamically manage topics based on the number of connected devices and their data types. This dynamic management allows the system to efficiently allocate resources and prioritize critical messages, ensuring timely delivery and processing even as the number of subscribers increases.
- **Scalable Architecture:** The architecture intends to be modular and scalable, enabling the incorporation of additional brokers and servers as required through the registration topic. This flexibility ensures that the system can grow to accommodate a wide range of subscribers and increased message volumes without requiring significant redesign.

These scalability features collectively ensure that the proposed system can effectively handle a growing number of subscribers and messages, maintaining performance and reliability in real-world applications.

3.3. Proposed Anatomy of SGP MQTT Topic

The categorization of topics has significant importance within the MQTT protocol due to its ability to provide streamlined and proficient communication, systematic data organization, enhanced security measures, and insightful analytics. Figure 4 illustrates the anatomical structure of both conventional MQTT packets and the proposed SGP packets.

In traditional MQTT communication (Section A of Figure 4), the inclusion of MQTT topics in every message sent introduces additional overhead to each communication. The potential impact of this overhead in systems with huge volumes might be substantial, resulting in heightened bandwidth consumption and diminished performance. The management of subscriptions to topics by MQTT brokers is a kind of overhead. When a customer opts to subscribe to a substantial quantity of subjects, the broker may experience a notable consumption of resources, resulting in a decline in performance. The repetition of levels occurs in each data packet sent at a frequency of one per second, as determined by the

farmer's selection, either from the client to the broker or in the other direction. These data packets consist of many subject levels, which are delimited by a forward slash (/). As seen in Figure 4, the components involved in this scenario include an actuator, temperature, and humidity sensors, a digital output, and an analog input. It is noteworthy to acknowledge that within each transmission, certain topic levels undergo value changes while others remain consistent. However, due to the nature of sending the entire data packet, topic levels that remain unchanged are still transmitted, resulting in an augmentation of the number of bytes and data volume. Consequently, this leads to a decrease in the performance of the MQTT protocol. According to [39], the citation is as follows: as seen in Figure 4, it is evident that there are some limitations. Specifically, only three parameters undergo modifications, while two parameters, namely, temperature and digital output, remain constant. However, by the conventional broker structure, these two fixed values are redundantly sent to the server on each occasion. Our proposed framework (Section B of Figure 4) integrates a unique topic for each data point and its corresponding value, encapsulating them as individual packets inside the MQTT broker. One benefit of this system is its ability to selectively transmit only relevant data to the server, namely, events and parameter changes occurring inside the greenhouse. This approach effectively reduces the transmission of extraneous information to the server. The example shown in section B of Figure 4 demonstrates the modification and transmission of three parameters, namely, actuator, humidity, and analog input, to the server. This is achieved without the need to transmit two fixed values.

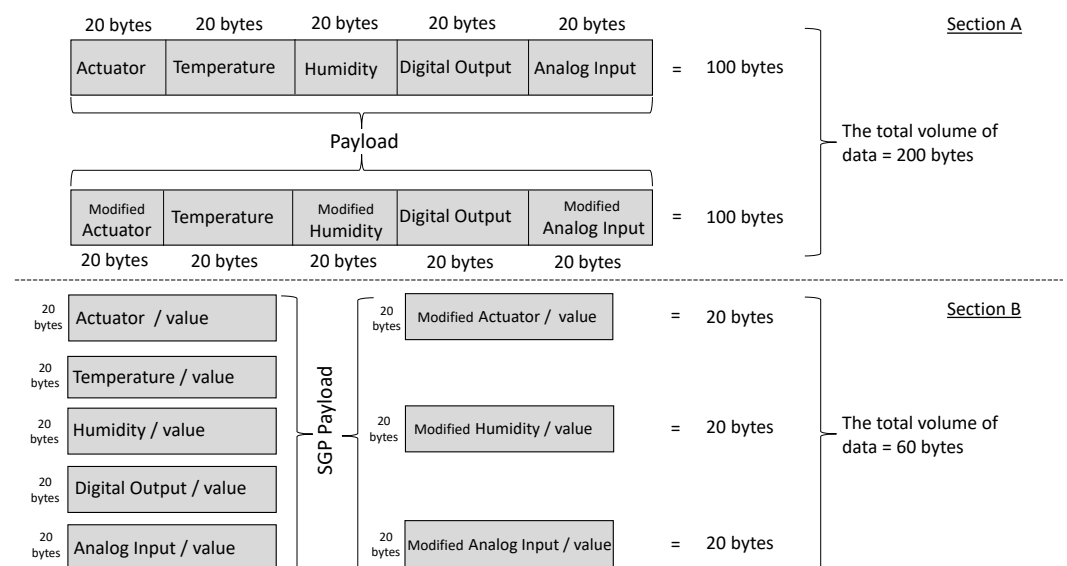


Figure 4. A comparison of two greenhouse data transmission payload frameworks; Section A: Traditional MQTT communication. Section B: Proposed MQTT communication.

In addition, it is possible to minimize the quantity of messages and the volume of data sent to a client in the form of a packet, resulting in a more scalable and efficient system when compared to section A. Consequently, the total data volume is reduced from 200 bytes to 60 bytes. In essence, the proposed topic standardization feature enables clients to use filters to sort communications according to certain themes, hence reducing the influx of extraneous information. Consequently, this approach effectively mitigates data redundancy. The appropriate categorization of topics facilitates the systematic arrangement of data, hence enhancing its manageability and maintenance. Additionally, it aids in the process of searching for and retrieving certain information. The critical factor is in the comprehensive evaluation of the advantages, whereby our suggested anatomical structure ultimately results in a decrease in latency for message transmission, hence requiring a reduced amount of traffic.

Our proposed architecture establishes a standard configuration method in MQTT for sending and receiving crucial greenhouse data and events. The introduced approach classifies sensor data into topic-specific categories, facilitating the control of actuators and ultimately reducing the message load on the primary server. It is imperative to conduct thorough compatibility assessments with commonly used hardware and software configurations in order to guarantee seamless integration into existing IoT ecosystems within intelligent greenhouse environments. This includes designing and integrating hardware components like microcontrollers and wireless communication modules to support the SGP MQTT protocol, enabling the control of actuators and acquisition of sensor data through output and input controllers such as relays and analog-to-digital converters. Software configuration guidelines are crucial for configuring MQTT brokers or implementing new ones to accommodate the structured message format of SGP, detailing necessary software libraries, APIs, or configuration settings. Optimizing data flow involves explaining how sensor data travel through the SGP MQTT protocol to reach the primary server or cloud platform efficiently, ensuring timely transmission of critical greenhouse data. Furthermore, ensuring interoperability with diverse IoT devices and platforms is paramount, emphasizing adherence to standards and compatibility across various vendor solutions.

4. Performance Evaluation

The objective of this chapter is to assess the effectiveness of the categorized topic structure we propose for the MQTT broker. This evaluation is conducted based on several key performance indicators, including pub-sub message size, the number of messages transmitted from the greenhouse gateway to the main server, the processing time for message transactions in the main server, the coverage rate of events, and the throughput of subscriptions.

- Pub-sub message size: About the topic matter, it is noteworthy that the SGP MQTT message arrangement necessitates the utilization of 20 bytes for transmitting each message about alterations in the greenhouse environment to a single server. In contrast, the traditional MQTT protocol consumes 100 bytes for the same purpose, as shown in Figure 4.
- The number of messages: This describes the overall capacity of a broker to effectively handle a particular number of messages during a given period. The significance of this measure is of utmost relevance within the realm of intelligent greenhouses, where the interchange and management of real-time data represent an important role in improving climate circumstances and efficiently using resources to enhance plant development [36,37,40].
- Message processing time: This refers to the amount of time it takes for the primary server to receive incoming messages, which has a direct influence on the system's capacity to stay synchronized with events taking place in the surrounding environment. The system's capacity to promptly and effectively address real-world occurrences is immediately impacted. By closely monitoring and strategically optimizing the processing times of messages, it is possible to guarantee that applications using the MQTT protocol can efficiently adapt to the ever-changing circumstances of the environment [34,41,42].
- Subscription throughput: The assessment of MQTT protocol performance is heavily reliant on the analysis of subscription throughput, as shown by previous studies conducted by various academics [16,28,43]. The term "message delivery rate" pertains to the speed at which a broker can transmit messages to its subscribers. The significance of this statistic lies in its direct influence on the responsiveness and efficiency of communication inside a system that utilizes the MQTT protocol. A high rate of subscription throughput signifies the MQTT broker's ability to effectively manage a large volume of subscribers and swiftly distribute messages to them [43,44]. Real-time transmission is of utmost importance in greenhouse applications [16].

Therefore, by the assertion made in this research, which pertains to the overall suitability of this methodology inside a greenhouse setting, we must substantiate and assess its efficacy. As a result, the findings of the assessment are compared to those of the standard MQTT and other current techniques.

4.1. Evaluation Assumptions

In the following analysis, we systematically determine each of the assumptions that were used to evaluate the effectiveness of the suggested categorized topic structure. Listed below are the assumptions used in this study:

- The MQTT brokers examined in this study are Mosquitto 1.4.15, Bevywise MQTT Route 2.0 (known for scalability), ActiveMQ 5.15.8, and EMQ X 4.0.8 (known for non-scalability). The performance of our suggested system was evaluated by using the values of messages in three different QoS zones of the MQTT protocol. This evaluation was conducted based on the analysis presented in the paper by [25].
- In one second, a process comprising four payloads is transmitted from the greenhouse gateway to the MQTT broker.
- In the conventional MQTT protocol, the structure of every payload is defined by five topic levels. Each topic level is allocated a data volume of 20 bytes, resulting in a total data volume consumption of 100 bytes for each payload. In contrast, the proposed structure of each MQTT payload designates its allocation to a distinct topic level, with a fixed data size of 20 bytes.
- The design proposal in our study receives environmental parameters and events from sensors located throughout the greenhouse. These data are relayed to an MQTT broker, which acts as a gateway for our greenhouse structure.

4.2. Evaluation Model

To evaluate the efficacy of the MQTT broker regarding message size and message quantity, it is necessary to use the $M/M/1$ queuing model as proposed by [45] in their study on performance analysis. In the standard $M/M/1$ model:

- The first “ M ” denotes that the arrival process follows a Poisson distribution, indicating that the time intervals between the arrival of messages are exponentially distributed.
- The second “ M ” signifies that the service process follows an exponential distribution, indicating that the time required for the broker to process a message is exponentially distributed.
- The numeral “1” in the system signifies the existence of a solitary server, namely the MQTT broker, according to [46].

In our analysis, we adapt the $M/M/1$ model to reflect the characteristics of MQTT protocol communication. The critical variables for our modified model are the message arrival rate (λ) and the service rate (μ), along with specific MQTT-related factors such as message size and QoS levels.

Modifications to the $M/M/1$ Model:

- Message Arrival Rate (λ^*): This represents the rate at which messages arrive at the MQTT broker. This rate varies depending on the QoS levels and the payload size. For different brokers and QoS levels, λ^* will differ.
- Message Size (π^*): Unlike the standard $M/M/1$ model, which may not account for varying message sizes, our modified model incorporates the fixed size of messages (π^*) for both the original and proposed MQTT brokers, measured in bytes.
- Service Rate (μ): The broker’s capacity to handle incoming messages determines the service rate. It reflects the processing capabilities of different MQTT brokers under various configurations.

Formalization: Let λ be the arrival rate of messages, and let μ be the service rate of the MQTT broker. Assuming that the arrival rate follows a Poisson distribution and the service time follows an exponential distribution, the system can be modeled as an $M/M/1$ queue, denoted by this abbreviation $q_{M/M/1} = (\lambda, \mu)$. Furthermore, the expected number of messages and the expected time a message spends in the system are computed employing (L) and (W) as shown in Equations (1)–(3).

$$L = \lambda / (\mu - \lambda) \quad (1)$$

$$W = 1 / (\mu - \lambda) \quad (2)$$

$$L = \lambda W \quad (3)$$

Importantly, as demonstrated in the assumption, we assume two variables in both the original and proposed MQTT protocol broker frameworks: the arrival message rate in three QoS levels, and the message size. According to Formulas (1)–(3), we provide our proposed Formula (4) for our architecture to obtain the expected number of messages through the $M/M/1$ model on the proposed MQTT broker architecture.

$$L^* = \lambda^* / \pi^* \quad (4)$$

- L^* represents the anticipated number of messages that are published in each topic.
- λ^* refers to the rate at which messages arrive, measured in messages per second.
- π^* represents the fixed size of messages for both the original and proposed MQTT brokers, measured in bytes.

By applying the modified $M/M/1$ model mentioned to the proposed categorized topics structure in MQTT, we can analyze the performance of the system in terms of the message size, the number of messages, and the message transaction process [45]. This analysis can help us optimize the performance of the MQTT broker and ensure that the system can handle the expected message load through the following strategies:

- **Optimized Topic Categorization:** As detailed in Section 3.2, we propose a standardized message structure with unique topics for each data point and event within the greenhouse environment. This approach reduces topic overlap and ensures efficient routing and processing of messages.
- **Payload Reduction:** According to Section 3.3, the payload of the proposed model is optimized to reduce the transmission of extra and irrelevant data by approximately 80 bytes compared to the traditional MQTT broker's payload framework. This significant reduction in payload size decreases network congestion and improves message processing efficiency.
- **Message Processing Time Optimization:** Leveraging the optimized $M/M/1$ queuing model, we focus on optimizing the message processing time (MPT), which has a direct impact on system capacity. The MPT is calculated by two factors: the number of events (E^*) and the arrival rate (λ). According to our assumption (four payloads are transmitted from greenhouse gateway to broker in one second), the MPT optimizes using the following formula:

$$MPT = \lambda^* / E^* \quad (5)$$

By adjusting the arrival rate depending on the payload size and ensuring efficient processing per event, we enhance the broker's capability to handle higher loads.

- **Dynamic Adjustment of Arrival and Service Rates:** The modified $M/M/1$ model allows for dynamic adjustments of the message arrival rate (λ) and service rate (μ) based on real-time data. This adaptability ensures the broker can maintain optimal performance under varying load conditions.
- **Efficient Message Handling:** Through the organization of messages into specific topics and the reduction of payload size, we optimize the efficiency of the message-handling process. This scenario enhances the broker's processing efficiency and boosts

its capacity to handle larger volumes of transactions, especially in situations with heavy traffic.

Likewise, relying on the assumptions indicated in Section 4.1, we separate the findings of our comparison in Tables 1–3 for simpler and superior comprehension. Table 1 indicates the number of messages transmitted in both broker configurations as estimated by the modified Formula (4) with the two primary parameters of message arrival rate and message size.

Table 1. Evaluation results for scalable and non-scalable MQTT brokers with various message arrival rates and average latency for QoS0.

The Number of Events = 4, QoS0								
MQTT Brokers	Original Mosquitto 1.4.15	Proposed Mosquitto 1.4.15	Original Bevywise MQTT Route 2.0	Proposed Bevywise MQTT Route 2.0	Original ActiveMQ 5.15.8	Proposed ActiveMQ 5.15.8	Original EMQX 4.0.8	Proposed EMQX 4.0.8
Message arrival rate (msgs/s)	32,016.00	32,016.00	32,839.00	32,839.00	39,479.00	39,479.00	18,034.00	18,034.00
Message size (bytes)	100	20	100	20	100	20	100	20
The number of messages	320.16	1600.8	328.39	1641.95	394.79	1973.95	180.34	901.7
Message processing time (s)	80.04	400.2	82.0975	365.4875	98.6975	493.4875	45.085	225.425
Average latency (in ms)	1.65	1.65	1.13	1.13	2.33	2.33	1.34	1.34
Subscription throughput (msgs/s)	194.036	970.181	290.610	1453.053	169.437	847.188	134.582	677.388

Table 2. Evaluation results for scalable and non-scalable MQTT brokers with various message arrival rates and average latency for QoS1.

The Number of Events = 4, QoS1								
MQTT Brokers	Original Mosquitto 1.4.15	Proposed Mosquitto 1.4.15	Original Bevywise MQTT Route 2.0	Proposed Bevywise MQTT Route 2.0	Original ActiveMQ 5.15.8	Proposed ActiveMQ 5.15.8	Original EMQX 4.0.8	Proposed EMQX 4.0.8
Message arrival rate (msgs/s)	9488.00	9488.00	3542.49	3542.49	12,873.00	12,873.00	4633.41	4633.41
Message size (bytes)	100	20	100	20	100	20	100	20
The number of messages	94.88	474.4	35.4249	177.1245	128.73	643.65	46.3341	231.6705
Message processing time (s)	23.72	118.6	8.856225	44.281125	32.1825	160.9125	11.583	57.917
Average latency (in ms)	0.74	0.74	0.96	0.96	1.38	1.38	0.87	0.87
Subscription throughput (msgs/s)	128.216	641.081	36.900	184.504	93.282	466.413	53.257	266.287

Table 2 displays the number of message transaction processes in one second for both the original and proposed broker mechanisms, based on the number of messages and events that occur per second in the greenhouse, which, in this investigation, is based on the premise of a constant value of 4. It is assumed that one package containing four events (which occur in a single second) is transferred from the greenhouse environment to the broker every second. Based on the assessment tables supplied in this subsection, we estimate that our proposed approach, i.e., a novel categorization for the topics accessible in the Mosquitto 1.4.15 broker, is compatible with the requirements of a smart greenhouse operated with IoT. Moreover, the subscription throughput in MQTT protocols refers to the ability of the MQTT broker to handle a large number of subscriptions or topic filters from multiple clients simultaneously, which is denoted by the abbreviation P^* . In other words, when a subscriber subscribes to a topic, the subscriber expresses an interest in receiving messages that are published on that topic. The MQTT broker, which is responsible for receiving and delivering messages, needs to be able to handle the traffic of messages that are published on the topic and deliver them to all the subscribers who have subscribed to that topic. Therefore, subscription throughput (P^*) is a critical factor in MQTT because it determines how quickly messages can be delivered to subscribers [16,28,43,44]. The subscription throughput in an MQTT broker can be calculated using the following Formula (6), where L^* is the number of messages subscribed per second, and T is the average latency of message delivery.

$$P^* = L^* / T \quad (6)$$

Table 3. Evaluation results for scalable and non-scalable MQTT brokers with various message arrival rates and average latency for QoS2.

The Number of Events = 4, QoS2								
MQTT Brokers	Original Mosquitto 1.4.15	Proposed Mosquitto 1.4.15	Original Bevywise MQTT Route 2.0	Proposed Bevywise MQTT Route 2.0	Original ActiveMQ 5.15.8	Proposed ActiveMQ 5.15.8	Original EMQ X 4.0.8	Proposed EMQX 4.0.8
Message arrival rate (msgs/s)	6585.00	6585.00	2649.00	2649.00	10,585.00	10,585.00	2627.31	2627.31
Message size (bytes)	100	20	100	20	100	20	100	20
The number of messages	65.85	329.25	26.49	132.45	105.85	529.25	26.2731	131.3655
Message processing time (s)	16.4625	82.3125	6.6225	33.1125	26.4625	132.3125	6.568	32.841
Average latency (in ms)	1.38	1.38	1.53	1.53	2.14	2.14	3.68	3.68
Subscription throughput (msgs/s)	47.717	238.58	17.313	86.568	49.462	247.313	7.1394	35.6971

4.3. Result

Within this particular section, we undertake an examination of the effects brought about by the traditional pub-sub broker messaging system. This analysis is conducted via the consideration of many noteworthy characteristics, and subsequently, we proceed to describe the performance of our proposed message system. This study aims to illustrate the findings obtained from the evaluation of non-scalable and scalable brokers, namely, Mosquitto 1.4.15, Bevywise MQTT Route 2.0, ActiveMQ 5.15.8, and EMQ X 4.0.8. The assessment is conducted on both the original and proposed designs. Subsequently, a comparative analysis is presented.

The brokers' performance is analyzed based on the previous calculations using four indicators:

- The rate at which messages are sent from the client to the broker per second is compared between the original and suggested broker systems.
- The measurement of the aggregate number of bytes sent during a given message exchange serves as a metric for evaluating the size of the message.
- The calculation of message processing time in four events occurring between the client and the brokers serves as an indication of the extent to which events are covered inside the greenhouse environment.
- The estimation of subscription throughput in MQTT brokers is a crucial indicator utilized to determine the efficiency of message transmission to subscribers.

Hence, the performance criterion indicators for the four protocols, including both the existing framework and our proposed structure, are delineated individually via five Figures 5–9. These charts specifically pertain to the three QoS levels of the brokers.

4.3.1. Message Processing Time Evaluation Results with the Number of Messages and Message Size

In this section, Figure 5 compares the message processing time through one second with the payload comprising four events depending on the number of messages delivered from the client in the greenhouse environment to the scalable and non-scalable brokers with the traditional and proposed mechanisms. Overall, what stands out from the data is that message processing time significantly improves approximately five-fold through all QoS categories for both scalable and non-scalable brokers, which means that our architecture enhances the protection of an IoT-equipped greenhouse system five times. Another noticeable point is that the Active MQ 5.15.8 scalable proposed broker takes on the most events coverage in the greenhouse environment across all QoS levels, whereas, in all QoS categories, another scalable proposed broker, EMQ X 4.0.8, has easily the lowest rates compared to non-scalable brokers.

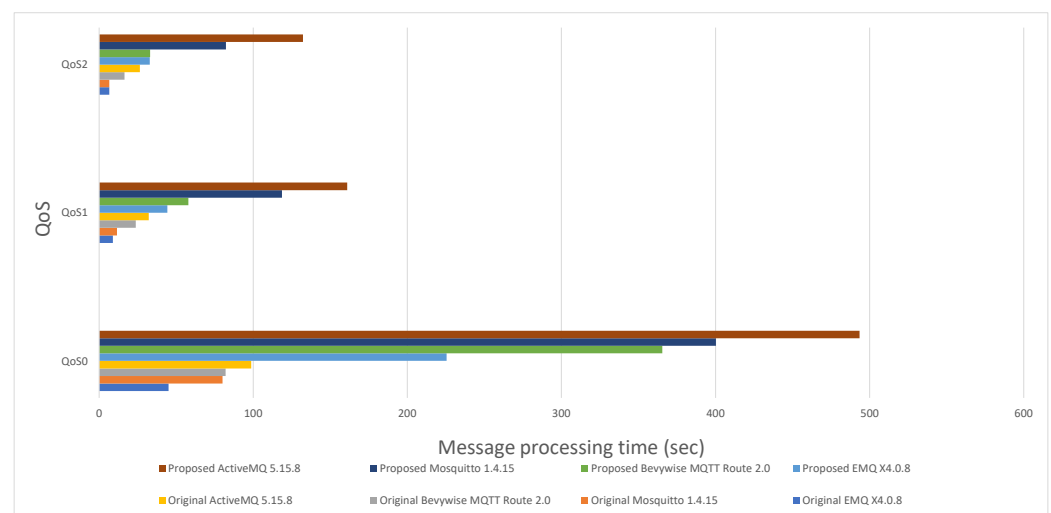


Figure 5. A comparison of the message processing time (sec) of all scalable and non-scalable brokers in both original and proposed architectures across all the QoS categories.

Looking at the details, as regards the original scalable 1.4.15 and Bevywise MQTT Route 2.0 brokers, their message processing time is calculated at 80.04 and 82.0975, respectively, which means the original Mosquitto broker covers 80.04 events and the original Bevywise MQTT Route 2.0 broker covers 82.0975 events in one second. By contrast, in our proposed method, the amount of information coverage for one second is 400.2 and 365.4875 in turn,

which makes all the events that happened in the greenhouse environment covered by the broker with our proposed topic classification.

The improved message processing time in our architecture signifies the following:

- Enhanced coverage and management of events, ensuring more events are processed promptly.
- Increased capability of the broker to manage a larger volume of concurrent events or messages.
- Improved responsiveness of the system to high event volumes, which is critical for real-time monitoring and control applications.

Figures 6 and 7 demonstrate the number of messages in both the original and proposed scalable and non-scalable brokers at all QoS levels. When we look at the message size or data volume that is reduced in the proposed firmware, the number of messages grows dramatically compared to both types of traditional MQTT brokers. For instance, if the two processes of temperature rise and declining humidity in the greenhouse environment are investigated, these two processes with our proposed architecture of brokers have much less data volumes or relatively small message sizes, much greater number of messages in one second, and, even according to Figure 5, more events covered per second. Looking at the specifics, the proposed ActiveMQ 5.15.8 broker has a sharp increase in the number of messages across all QoS levels. Furthermore, there is a gradual rise in the proposed Mosquitto 1.4.15 broker through QoS1 and 2 categories, while the results of the proposed Bevywise MQTT Route 2.0 broker have a considerable jump at the QoS1 level.

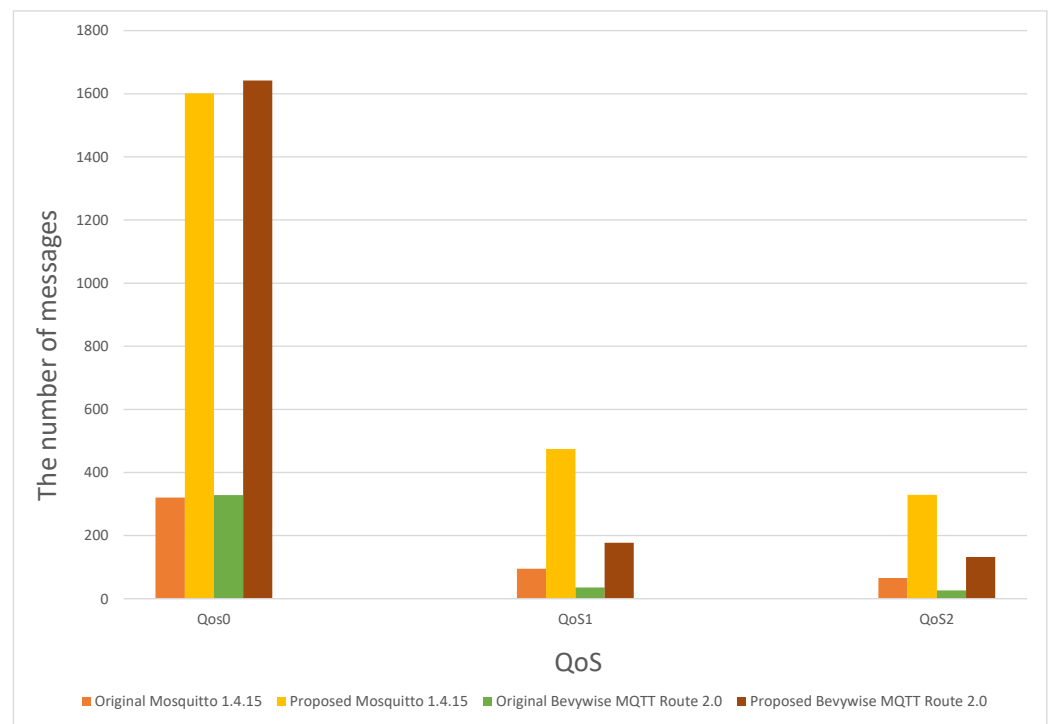


Figure 6. The number of messages from non-scalable brokers in both original and proposed architectures across all QoS categories.

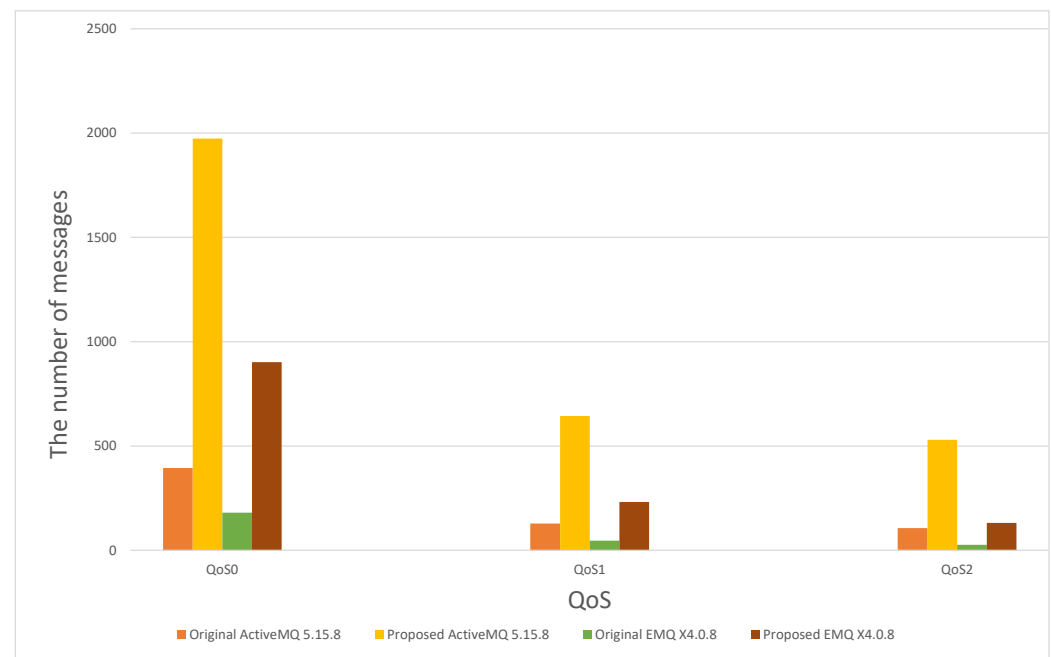


Figure 7. The number of messages from scalable brokers in both original and proposed architectures across all QoS categories.

4.3.2. Subscription Throughput Evaluation Results with the Number of Messages

Figures 8 and 9 show the subscription throughput of non-scalable and scalable brokers in both traditional and proposed architecture at all QoS levels, respectively. There are significant changes in the results of our proposed structure for non-scalable brokers by three QoS levels. See Figure 8. Another interesting point is that the proposed Bevywise MQTT Route 2.0 broker has the highest amount of throughput only at the QoS level, while the other broker sees an upward trend at the QoS1 and 2 levels. This means that brokers with higher throughput results subscribe the messages to the brokers at a higher speed and more efficiency. Regarding Figure 9, the proposed ActiveMQ 5.15.8 broker grows more steeply than the proposed EMQ X 4.0.8 at all QoS categories. Meanwhile, the high throughput value of brokers utilizing our proposed architecture compared to brokers with the traditional architecture indicates that brokers can manage a large number of messages simultaneously and ensure that messages are delivered quickly. In addition, a high throughput improves reliability and scalability by effectively addressing message traffic and minimizing the probability of missing or lost messages. In other words, the sign of a high subscription throughput is that the MQTT broker can quickly and efficiently route messages to the appropriate subscribers without impacting the overall performance of the system. This ensures that messages are delivered in a timely and reliable manner, which is crucial for many IoT applications [43,44].

Sorting the proposed MQTT brokers based on the subscription throughput (from highest to lowest: left to right). At QoS0: Bevywise MQTT Route 2.0, Mosquitto 1.4.15, ActiveMQ 5.15.8, EMQ X 4.0.8. At QoS1: Mosquitto 1.4.15, ActiveMQ 5.15.8, EMQ X 4.0.8, Bevywise MQTT Route 2.0. At QoS2: ActiveMQ 5.15.8, Mosquitto 1.4.15, Bevywise MQTT Route 2.0, EMQ X 4.0.8.

Sorting all brokers according to the message processing time based on Figure 5 (from highest to lowest: bottom to top). At QoS0: ActiveMQ 5.15.8, Mosquitto 1.4.15, Bevywise MQTT Route 2.0, EMQ X 4.0.8. At QoS1: ActiveMQ 5.15.8, Mosquitto 1.4.15, EMQ X 4.0.8, Bevywise MQTT Route 2.0. At QoS2: ActiveMQ 5.15.8, Mosquitto 1.4.15, Bevywise MQTT Route 2.0, EMQ X 4.0.8.

Sorting both scalable and non-scalable brokers with our proposed mechanism based on the number of messages (from highest to lowest: right to left). At QoS0: ActiveMQ 5.15.8, Bevywise MQTT Route 2.0, Mosquitto 1.4.15, EMQ X 4.0.8. At QoS1: ActiveMQ

5.15.8, Mosquitto 1.4.15, EMQ X 4.0.8, Bevywise MQTT Route 2.0. At QoS2: ActiveMQ 5.15.8, Mosquitto 1.4.15, Bevywise MQTT Route 2.0, EMQ X 4.0.8.

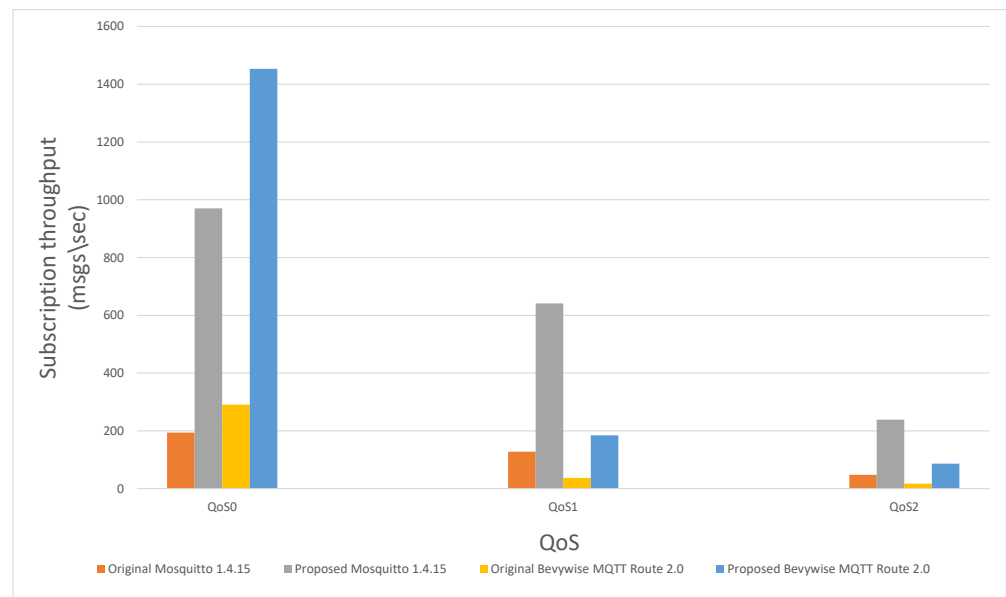


Figure 8. Subscription throughput (msgs/s) of non-scalable brokers in both the original and proposed architectures across all QoS levels.

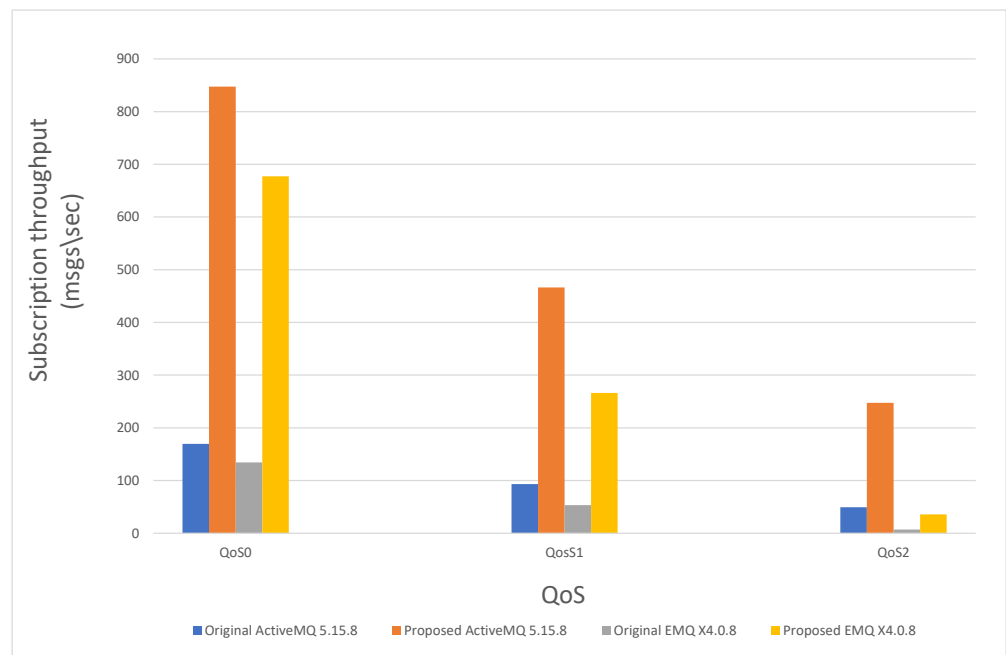


Figure 9. Subscription throughput (msgs/s) of scalable brokers in both the original and proposed architectures across all QoS levels.

To summarize our evaluation experiments, we can state that ActiveMQ scales well to beat all other brokers' performance on our proposed pub-sub mechanism and topic categorizations in a greenhouse environment. It is the most reliable scalable broker examination we have ever assessed. EMQ X also performs well with our proposed architecture for the greenhouse environment. On the other hand, if the hardware is resource-constrained (Performance), then Mosquitto or Bevywise MQTT Route can be taken as better choices over non-scalable brokers [25].

4.4. Comparison with Existing Solutions

This section presents a comparative analysis of the scalability, reliability, complexity, and server processing capabilities of our proposed standardized MQTT broker-equipped smart greenhouse protocol with alternative MQTT broker designs that are deemed adequate for the agricultural setting. Table 4 provides a comprehensive overview of the comparison between our recommended design and the corresponding attributes of comparable research discussed in the literature, as previously elucidated in Section 2. The primary objective of the comparison presented in Table 4 is to demonstrate that our proposed framework possesses the characteristics found in different categories of IoT-enabled smart greenhouses. These features are achieved by utilizing the standardized MQTT protocol [4,27,36] and facilitating the transmission of all environmental events to the central server in an organized manner through the implementation of topic categorization and a pub-sub system. The MQTT broker architectures presented in references [28,34,36] lack standardization for topics, and their broker architecture closely resembles the conventional MQTT broker seen in Figure 4. As a result, these systems are unable to provide the functionalities of efficient server processing, seamless client-to-broker data transmission, and a pub-sub architecture, hence presenting significant challenges [4,23,24]. The approach we offer exhibits a standardized pattern that is tailored to the unique characteristics of the greenhouse environment. This pattern effectively reduces the computational load on the server and enhances its resilience against large-scale data processing. Furthermore, about scalability, it can be seen that the plans [28,34,36] exhibit a deficiency in scalability, accompanied by a lower level of dependability in their architectural design. The architecture they developed is not suitable for real-time applications, such as the monitoring of crucial greenhouse variables. In the context of greenhouse management, it is important to address certain scenarios such as device upgrades, sensor additions, and critical plant conditions. These situations may lead to a lack of notification to the server, as the relevant topics may not be adequately covered. However, our proposed standard architecture offers a solution to this issue. By employing a reliable pub-sub mechanism and implementing topic classification, any decision made during greenhouse product monitoring can be effectively executed.

Table 4. Comparison of computational complexity and resource requirements for MQTT broker categorization.

Solution	Topic Hierarchy/Broker	Pub-Sub Mechanism	Message Size	Distribution
Our proposal	9 topics/1 broker	Simple	Low	Very high scalability, very low complexity, very high reliability, low server processing, suitable for smart greenhouse
[4]	3 topics/1 broker	Medium complex	Medium	Medium scalability, high complexity, medium reliability, heavy server processing, suitable for smart greenhouse
[27]	3 topics/1 broker	Medium complex	Medium	Medium scalability, high complexity, medium reliability, heavy server processing, suitable for smart greenhouse
[28]	None/5 brokers	High complex	High	Non-scalable, very high complexity, very low reliability, very high server processing
[30]	1 topic/M-brokers	High complex	High	Very low scalability, high complexity, very low reliability, very high server processing
[34]	None/5 brokers	High complex	High	Non-scalable, high complexity, very low reliability, heavy server processing
[36]	None/1 broker	Complex	Very High	Non-scalable, very high complexity, very low reliability, very heavy server processing

However, previous studies conducted by researchers [4,27] have classified topics in a way that only supports a portion of the accessible data in the greenhouse. These classifications do not include all the events linked to various components, such as the connection status and the process of updating the server based on alterations made in the greenhouse. This section examines the benefits associated with the smallest message size within each level of a broadcast payload, in contrast to the findings provided in Table 4. The design we propose effectively mitigates the issue of data redundancy and minimizes data transmission, resulting in improved efficiency in broker processing time and reduced network congestion.

Based on the aforementioned assessment, it can be concluded that the proposed pub-sub mechanism, which is deemed dependable, and the standardized MQTT-based SGP exhibit exceptional scalability and dependability. This is achieved via a straightforward design and minimal server processing requirements.

Various MQTT optimization strategies assess their computational complexity and resource requirements by considering the number of topics, events, message size, and subscription throughput as seen in Table 5.

Table 5. Quantitative comparison of MQTT optimization techniques.

Solution	Protocols	Number of Topics	Number of Events	Message Size (Bytes)	Subscription Throughput (msgs/s)
Our proposal	MQTT	9	4	20	677.388–1453.053
[11]	Seven messaging and data communication protocols	1	4	100	80–480
[47]	MQTT	3	10	73	-
[4]	MQTT, AMQP, DSS	3	2	32	-

The article [11] utilized seven distinct messaging and data communication protocols; it used one topic, four events, and a message size of 100 bytes, with a subscription throughput of 80–480 msgs/s. The single topic with 100 byte messages suggests higher processing and memory usage. The lower throughput compared to our method indicates less efficiency under high load. The article [47] used MQTT, with three topics, ten events, and a message size of 73 bytes; the subscription throughput was not specified. This setup has medium complexity with three topics, and the moderate message size reduces the resource load compared to the first article. Nevertheless, the absence of throughput data presents a challenge in evaluating efficacy under real-world conditions. The article [4] utilized MQTT, AMQP, and DSS, with three topics, two events, and a message size of 32 bytes, but the subscription throughput was not specified. Although this study defines three different topics, it fails to provide the same level of specificity and granularity as our approach, which encompasses nine topics. This discrepancy in the number of topics can affect the system's capacity to efficiently manage and categorize various categories of data, thereby influencing the overall performance and complexity.

Our proposed approach employs both scalable and non-scalable MQTT brokers, with nine topics, four events, a message size of 20 bytes, and a subscription throughput of 677.388–1453.053 msgs/s. This approach has a lower complexity due to a hierarchical and dynamic topic management system, significantly reduced resource requirements due to smaller message size, and the highest throughput among the compared methods, indicating superior efficiency and scalability.

Quantitative comparisons show that our method has the smallest message size (20 bytes), which leads to reduced computational load and memory usage, which is crucial for large-

scale and high-frequency data environments like smart greenhouses. Regarding subscription throughput, our approach (677.388–1453.053 msgs/s) demonstrates the highest capacity, indicating that it can handle a significantly higher number of messages per second than the first article (80–480 msgs/s). This makes our method more suitable for real-time applications requiring rapid data processing and high scalability.

5. Conclusions and Future Work

This article presents and assesses a reliable pub-sub mechanism and a standardized topic structure that exhibits significant scalability for the protocol utilized in IoT-enabled smart greenhouses. The proposed topic structure facilitates effective communication and comprehensive coverage of any modifications taking place within the various components of the greenhouse environment, hence resulting in enhanced plant monitoring and management. The proposed pub-sub system aims to optimize the utilization of resources in the MQTT broker architecture by minimizing the number of bytes per topic, message load, and overall data volume. This study emphasizes the significance of reducing the overall volume of transmitted data, complexity, cost, and server processing in a modified MQTT broker. The objective is to improve the scalability, simplicity, and efficiency of the system by minimizing the reception of irrelevant message data and reducing data duplication. This is achieved through the assignment of data to their respective topics. The proposed communication architecture does not need the customization of the brokers, namely, Mosquitto 1.4.15, Bevywise MQTT Route 2.0, ActiveMQ 5.15.8, and EMQ X 4.0.8. Therefore, it is feasible to employ any existing broker to transmit messages with a standardized format from the greenhouse to the primary server. The assessment demonstrates that our system is capable of efficiently handling a substantial number of subscribers and promptly disseminating messages to them via the subscription key factor with high throughput. This factor directly influences the system's responsiveness and communication efficiency.

To ascertain the extent of our contribution, we evaluated our proposed mechanism across several MQTT brokers and performed a comparative analysis with relevant papers in the field. The process of categorizing topics enables the collection and analysis of data purposefully. This capability has the potential to be advantageous in the monitoring and enhancement of system performance.

Despite the promising results, this study has several limitations that need to be addressed to understand the full potential and limitations of the proposed method in real-world applications. While the proposed design improves scalability and efficiency, its primary focus is on MQTT brokers. Subsequent research should investigate the incorporation of other IoT communication protocols, such as CoAP, to ascertain their compatibility and performance in comparable applications. Likewise, the present system's dependence on pre-existing MQTT brokers, not customization, may restrict its capacity to use sophisticated functionalities that have become exclusive to certain broker implementations. Subsequent studies should explore the potential advantages of tailoring brokers to enhance performance optimization. The research did not investigate the security implications of the suggested design. Considering the growing significance of cybersecurity in IoT systems, future endeavors need to tackle possible weaknesses and integrate strong security measures to protect the data and infrastructure.

Author Contributions: Conceptualization, B.M.; Methodology, B.M.; Validation, B.V.; Formal analysis, B.M.; Investigation, B.M.; Resources, B.V.; Writing—original draft, B.M.; Writing—review & editing, B.V.; Supervision, B.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available in the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

- Maraveas, C.; Piromalis, D.; Arvanitis, K.; Bartzanas, T.; Loukatos, D. Applications of IoT for optimized greenhouse environment and resources management. *Comput. Electron. Agric.* **2022**, *198*, 106993. [\[CrossRef\]](#)
- Bersani, C.; Ruggiero, C.; Sacile, R.; Soussi, A.; Zero, E. Internet of things approaches for monitoring and control of smart greenhouses in industry 4.0. *Energies* **2022**, *15*, 3834. [\[CrossRef\]](#)
- Li, R.a.; Sha, X.; Lin, K. Smart greenhouse: A real-time mobile intelligent monitoring system based on WSN. In Proceedings of the 2014 international wireless communications and mobile computing conference (IWCMC), Nicosia, Cyprus, 4–8 August 2014; pp. 1152–1156.
- Ferraz Junior, N.; Silva, A.A.; Guelfi, A.E.; Kofuji, S.T. Performance evaluation of publish-subscribe systems in IoT using energy-efficient and context-aware secure messages. *J. Cloud Comput.* **2022**, *11*, 1–17. [\[CrossRef\]](#) [\[PubMed\]](#)
- Li, X.; Deng, Y.; Ding, L. Study on precision agriculture monitoring framework based on WSN. In Proceedings of the 2008 2nd International Conference on Anti-counterfeiting, Security and Identification, Guiyang, China, 20–23 August 2008; pp. 182–185.
- Asghari, P.; Rahmani, A.M.; Javadi, H.H.S. Internet of Things applications: A systematic review. *Comput. Netw.* **2019**, *148*, 241–261. [\[CrossRef\]](#)
- Maraveas, C. Incorporating artificial intelligence technology in smart greenhouses: Current State of the Art. *Appl. Sci.* **2022**, *13*, 14. [\[CrossRef\]](#)
- Mohamed, N.; Al-Jaroodi, J.; Jawhar, I.; Lazarova-Molnar, S.; Mahmoud, S. SmartCityWare: A service-oriented middleware for cloud and fog enabled smart city services. *IEEE Access* **2017**, *5*, 17576–17588. [\[CrossRef\]](#)
- Anagnostopoulos, T.; Zaslavsky, A.; Kolomvatsos, K.; Medvedev, A.; Amirian, P.; Morley, J.; Hadjieftymiades, S. Challenges and opportunities of waste management in IoT-enabled smart cities: A survey. *IEEE Trans. Sustain. Comput.* **2017**, *2*, 275–289. [\[CrossRef\]](#)
- Lohitha, N.S.; Pounambal, M. Integrated publish/subscribe and push-pull method for cloud based IoT framework for real time data processing. *Meas. Sens.* **2023**, *27*, 100699. [\[CrossRef\]](#)
- Lazidis, A.; Tsakos, K.; Petrakis, E.G. Publish-Subscribe approaches for the IoT and the cloud: Functional and performance evaluation of open-source systems. *Internet Things* **2022**, *19*, 100538. [\[CrossRef\]](#)
- Uslu, B.Ç.; Okay, E.; Dursun, E. Analysis of factors affecting IoT-based smart hospital design. *J. Cloud Comput.* **2020**, *9*, 1–23.
- Lin, J.; Yu, W.; Zhang, N.; Yang, X.; Zhang, H.; Zhao, W. A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications. *IEEE Internet Things J.* **2017**, *4*, 1125–1142. [\[CrossRef\]](#)
- Chakravarthy, R. An experimental study of IoT-Based topologies on MQTT protocol for agriculture intrusion detection. *Meas. Sens.* **2022**, *24*, 100470.
- Bacco, M.; Boero, L.; Cassará, P.; Colucci, M.; Gotta, A.; Marchese, M.; Patrone, F. IoT applications and services in space information networks. *IEEE Wirel. Commun.* **2019**, *26*, 31–37. [\[CrossRef\]](#)
- Glaroudis, D.; Iossifides, A.; Chatzimisios, P. Survey, comparison and research challenges of IoT application protocols for smart farming. *Comput. Netw.* **2020**, *168*, 107037. [\[CrossRef\]](#)
- Araujo, V.; Mitra, K.; Saguna, S.; Åhlund, C. Performance evaluation of FIWARE: A cloud-based IoT platform for smart cities. *J. Parallel Distrib. Comput.* **2019**, *132*, 250–261. [\[CrossRef\]](#)
- Du, X.; Zhou, Z.; Zhang, Y.; Rahman, T. Energy-efficient sensory data gathering based on compressed sensing in IoT networks. *J. Cloud Comput.* **2020**, *9*, 1–16. [\[CrossRef\]](#)
- Roy, D.G.; Mahato, B.; De, D.; Buyya, R. Application-aware end-to-end delay and message loss estimation in Internet of Things (IoT)—MQTT-SN protocols. *Future Gener. Comput. Syst.* **2018**, *89*, 300–316.
- Khaled, A.E.; Helal, A.; Lindquist, W.; Lee, C. IoT-DDL—device description language for the “T” in IoT. *IEEE Access* **2018**, *6*, 24048–24063. [\[CrossRef\]](#)
- Khaled, A.E.; Helal, S. Interoperable communication framework for bridging RESTful and topic-based communication in IoT. *Future Gener. Comput. Syst.* **2019**, *92*, 628–643. [\[CrossRef\]](#)
- Kim, G.; Kang, S.; Park, J.; Chung, K. An MQTT-based context-aware autonomous system in oneM2M architecture. *IEEE Internet Things J.* **2019**, *6*, 8519–8528. [\[CrossRef\]](#)
- Happ, D.; Wolisz, A. Limitations of the Pub/Sub pattern for cloud based IoT and their implications. In Proceedings of the 2016 Cloudification of the Internet of Things (CIoT), Paris, France, 23–25 November 2016; pp. 1–6.
- Rizzardi, A.; Sicari, S.; Miorandi, D.; Coen-Porisini, A. AUPS: An open source AUTHenticated Publish/Subscribe system for the Internet of Things. *Inf. Syst.* **2016**, *62*, 29–41. [\[CrossRef\]](#)
- Mishra, B.; Mishra, B.; Kertesz, A. Stress-testing MQTT brokers: A comparative analysis of performance measurements. *Energies* **2021**, *14*, 5817. [\[CrossRef\]](#)
- Uroz, D.; Rodríguez, R.J. Characterization and evaluation of IoT protocols for data exfiltration. *IEEE Internet Things J.* **2022**, *9*, 19062–19072. [\[CrossRef\]](#)
- Amanlou, S.; Hasan, M.K.; Bakar, K.A.A. Lightweight and secure authentication scheme for IoT network based on publish-subscribe fog computing model. *Comput. Netw.* **2021**, *199*, 108465. [\[CrossRef\]](#)
- Longo, E.; Redondi, A.E.C.; Cesana, M.; Manzoni, P. BORDER: A benchmarking framework for distributed MQTT brokers. *IEEE Internet Things J.* **2022**, *9*, 17728–17740. [\[CrossRef\]](#)
- Azzedin, F.; Alhazmi, T. Secure data distribution architecture in IoT using MQTT. *Appl. Sci.* **2023**, *13*, 2515. [\[CrossRef\]](#)

30. Amoretti, M.; Pecori, R.; Protskaya, Y.; Veltri, L.; Zanichelli, F. A scalable and secure publish/subscribe-based framework for industrial IoT. *IEEE Trans. Ind. Inform.* **2020**, *17*, 3815–3825. [\[CrossRef\]](#)
31. Fawwaz, D.Z.; Chung, S.H.; Ahn, C.W.; Kim, W.S. Optimal distributed MQTT broker and services placement for SDN-edge based smart city architecture. *Sensors* **2022**, *22*, 3431. [\[CrossRef\]](#)
32. Manzoni, P.; Maniezzo, V.; Boschetti, M.A. Modeling Distributed MQTT Systems Using Multicommodity Flow Analysis. *Electronics* **2022**, *11*, 1498. [\[CrossRef\]](#)
33. Mishra, B.; Kertesz, A. The use of MQTT in M2M and IoT systems: A survey. *IEEE Access* **2020**, *8*, 201071–201086. [\[CrossRef\]](#)
34. Veeramanikandan, M.; Sankaranarayanan, S. Publish/subscribe based multi-tier edge computational model in Internet of Things for latency reduction. *J. Parallel Distrib. Comput.* **2019**, *127*, 18–27.
35. Bello, O.; Zeadally, S. Toward efficient smartification of the Internet of Things (IoT) services. *Future Gener. Comput. Syst.* **2019**, *92*, 663–673. [\[CrossRef\]](#)
36. Fathy, C.; Ali, H.M. A secure IoT-based irrigation system for precision agriculture using the expeditious cipher. *Sensors* **2023**, *23*, 2091. [\[CrossRef\]](#) [\[PubMed\]](#)
37. Tzerakis, K.; Psarras, G.; Kourgialas, N.N. Developing an Open-Source IoT Platform for Optimal Irrigation Scheduling and Decision-Making: Implementation at Olive Grove Parcels. *Water* **2023**, *15*, 1739. [\[CrossRef\]](#)
38. Koroniotis, N.; Moustafa, N.; Sitnikova, E.; Turnbull, B. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. *Future Gener. Comput. Syst.* **2019**, *100*, 779–796. [\[CrossRef\]](#)
39. Tantitharanukul, N.; Osathanunkul, K.; Hantrakul, K.; Pramokchon, P.; Khoenkaw, P. Mqtt-topic naming criteria of open data for smart cities. In Proceedings of the 2016 International Computer Science and Engineering Conference (ICSEC), Chiang Mai, Thailand, 14–17 December 2016; pp. 1–6.
40. Donzia, S.K.Y.; Kim, H.K. Architecture Design of a smart farm system based on big data appliance machine learning. In Proceedings of the 2020 20th International Conference on Computational Science and Its Applications (ICCSA), Cagliari, Italy, 1–4 July 2020; pp. 45–52.
41. Nam, J.; Jun, Y.; Choi, M. High Performance IoT Cloud Computing Framework Using Pub/Sub Techniques. *Appl. Sci.* **2022**, *12*, 11009. [\[CrossRef\]](#)
42. Seoane, V.; Garcia-Rubio, C.; Almenares, F.; Campo, C. Performance evaluation of CoAP and MQTT with security support for IoT environments. *Comput. Netw.* **2021**, *197*, 108338. [\[CrossRef\]](#)
43. Banno, R.; Ohsawa, K.; Kitagawa, Y.; Takada, T.; Yoshizawa, T. Measuring performance of MQTT v5. 0 brokers with MQTTLoader. In Proceedings of the 2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 9–12 January 2021; pp. 1–2.
44. Tejashree, V.; Vidhyashree, N.; Anusha, S.; Anu, K.; Akshatha, P.; Kumar, S.D. Mqtt-sn based architecture for estimating delay and throughput in iot. In Proceedings of the Data Science and Computational Intelligence: Sixteenth International Conference on Information Processing (ICInPro 2021), Bengaluru, India, 22–24 October 2021; Proceedings 16; Springer: Berlin/Heidelberg, Germany, 2021; pp. 481–490.
45. Bouloukakis, G.; Moscholios, I.; Georgantas, N.; Issarny, V. Performance analysis of internet of things interactions via simulation-based queueing models. *Future Internet* **2021**, *13*, 87. [\[CrossRef\]](#)
46. Hmissi, F.; Ouni, S. TD-MQTT: Transparent Distributed MQTT Brokers for Horizontal IoT Applications. In Proceedings of the 2022 IEEE 9th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), Hammamet, Tunisia, 28–30 May 2022; pp. 479–486.
47. Vrettos, G.; Logaras, E.; Kalligeros, E. Towards standardization of MQTT-alert-based sensor networks: Protocol structures formalization and low-end node security. In Proceedings of the 2018 IEEE 13th International Symposium on Industrial Embedded Systems (SIES), Graz, Austria, 6–8 June 2018; pp. 1–4.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.