IoT Communication Based on MQTT and OneNET Cloud Platform in Big Data Environment

Youcheng Shan *, Yuening Su , Jiahui Lin , Tingyu Shan

Posted Date: 17 January 2024

doi: 10.20944/preprints202401.1250.v1

Keywords: Big data environment; IoT; MQTT; OneNET Cloud platform

Article

IoT Communication Based on MQTT and OneNET Cloud Platform in Big Data Environment

Youcheng Shan [1,*], Yuening Su [2], Jiahui Lin [3], Tingyu Shan [4]

[1] School of Logistics and Supply Chain Management, Zhejiang Technical Institute of Economics, Hangzhou, Zhejiang, China, 310018; 260046@zjtieledu.cn
[2] School of Electronic Information, Suzhou University, Suzhou, Jiangsu, China, 215031; 2028410189l@suda.edu.cn
[3] Material and Nanoscience, University of Waterloo, Waterloo, Ontarial, Canada, N2L 3G1; j384lin@uwaterloo.ca
[4] Mathematical Economics, University of Waterloo,Waterloo, Ontarial, Canada, N2L 3G1; t5shan@uwaterloo.ca
* Correspondence: 260046@zjtie.edu.cn

Abstract: The MQTT IoT connection technology is widely used, becoming the primary means for IoT devices to communicate with servers amid the vigorous development of IoT technology. On the OneNET cloud platform, MQTT protocol connection and data communication, message sending, and subscription are established, one-to-many message transmission is solved, and the reverse control from server to device is realized in big data environment. This technology can provide fast and secure messaging services for remote devices with a small code capacity and limited bandwidth. Experiments indicated that the communication mode has low cost, small bandwidth, and slight delay, which meets the requirements of device data acquisition and control in a large area and has broad application prospects and reference significance in the field of mobile applications.

Keywords: 1. Big data environment; 2. IoT; 3. MQTT; 4. OneNET Cloud platform

1. Introduction

With the gradual development of communication technology, a variety of communication technologies have emerged. HTTP is the representative of the request-response mode. HTTP/2 protocol also introduces the asynchronous request-response mode, the client can set different priorities for the request, and the server can decide which request to respond first according to the priority. In the case of HTTP, the re-establishment of a connection for each new request message results in significant overhead. The CoAP (Constrained Application Protocol) protocol is designed to be lightweight and can be used for resource constrained IoT devices. The LwM2M protocol is defined above the CoAP protocol, but it processes a step further on message transmission. It standardizes the device model based on IPSO (IP-based Smart Object), provides a set of lightweight device management and interactive interface protocols. The MQTT protocol is well suited to remote devices with limited computing power, low network bandwidth, unstable signal, and low power consumption. Meanwhile, the persistent connection used by MQTT can greatly reduce the overhead of re-connection. It gradually became the network protocol standardized for IoT communication system.

This paper proposes a combination scheme based on MQTT protocol and OneNET cloud platform considering the problems of data transmission, security, management and cost existing in the IoT communication technologies in recent years. The combination of the two models has emerged in the IoT applications recently and this scheme can effectively improve some of the problems encountered in the IoT communication, such as the slow delivery and reception speed of messaging, the real-time capabilities, the update management of message and data, etcetera. The MQTT

 doi:10.20944/preprints202401.1250.v1

(Message Query Transmission Transport) is a protocol under ISO / IECPRF 20922, based on the publish-subscribe messaging that uses TCP protocol [1]. For the devices with low hardware performance, the communication performance is in positive quality after joining the message broker. Currently, the IoT communication technologies have entered a new phase.

In the industrial applications of IoT, it is crucial to consider the measurement and control requirements for data type, size, and transmission rate based on big data [2–4]. Under such constraints, a targeted transmission protocol becomes essential.

2. Background and Principles

At present, the main domestic IoT cloud service manufacturers, Ali Cloud, Tencent Cloud, Baidu Cloud, OneNET cloud platform and so on, have been able to provide platform to realize MQTT protocol analysis and application. The MQTT protocol can be widely used in the industrial production field. The MQTT is designed as a lightweight, simple, and open transport protocol for easy implementation, based on the TCP / IP protocol [5–9].

Based on the MQTT protocol communication network technology, it can perform on the platform of other network protocols, which makes it quite easier to transmit data. It can be in progress orderly and ensure good bi-directional communication. The main benefit of this technology is the ability to provide authentic message support to remote users with less code and lower bandwidth. The construction principle is shown in Figure 1.

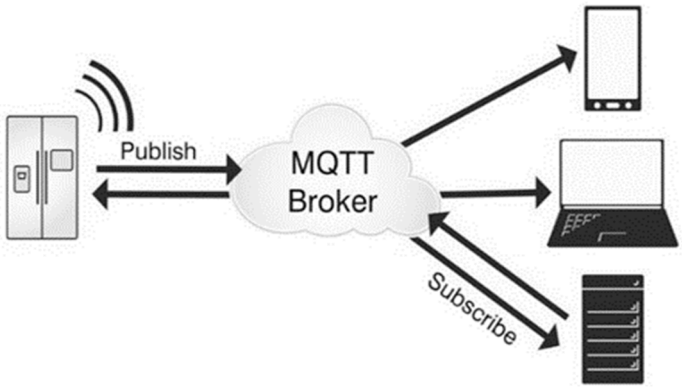

Figure 1. The MQTT architecture.

MQTT is a publish-subscribe based messaging protocol via client-server mode [10–12]. The technology is applicable in many cases, including in certain restricted environments, such as “machine to machine” communication and the IoT. It was widely used in satellite communication sensors, medical devices, smart-home devices, and certain small devices.

Connect with the OneNET cloud platform via this technology. OneNET, based on the structure of “cloud, network, edge and port”, has popular features including extensive computing, smart memory block loading, and APIs for typical IoT application scenarios. With OneNet, the architecture of IoT project is clear, elements such as the device, API, application, event trigger, and dataflow are organized by customer projects, and the input for every project is designed to be event triggered and derived by preset statements. OneNet is proven to be a cost-effective solution for fast-track IoT application development or concept verification [6,12].

This platform can provide developers with more possibilities to connect all intelligent hardware, and smart home products together, as well as providing comprehensive solutions for intelligent hardware and smart home products in a big data environment. Furthermore, it can also provide an efficient and low-cost hosting service, multi-protocol intelligent analysis, reliability and security in data storage and analysis, and multi-dimensional support, that is, immediacy and sustainability advantages.

 doi:10.20944/preprints202401.1250.v1

2.1. MQTT Communication Principle

MQTT will build the basic network transmission: it creates an orderly, non-destructive, byte-based bi-directional transmission between the client and the server [7]. The MQTT links the quality of service (QoS) to the topic when transmitting the application data.

Implementation requirements for MQTT: clients and servers. The MQTT protocol can be divided into three categories: publishers, brokers, and subscribers. In this case, the publisher and subscription are the client, while the publisher is the server, and the publisher can be the subscriber. The message sent by MQTT is divided into two parts: topic and payload. Topic, which means the type of message, will receive message about the topic after the subscriber orders it. Payload, which could be interpreted as message content, means what will be used specifically by users. The schematic diagram of the MQTT is shown in Figure 2.

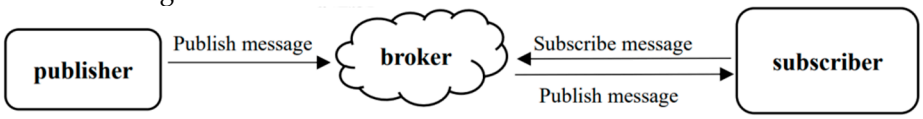

Figure 2. Schematic diagram of the MQTT.

The concrete content of the message is included in the payload of MQTT protocol. The message content is well readable in JSON sequence format which easy to be analyze and generated by machines, therefore, it is encoded in JSON format generally. In the IoT environment, different information types are defined to complete various functions, including device registration, acquiring gateway establishment equipment, setting up data collection tasks and instructions, etc. Any application or system, by the MQTT protocol, commonly has a network connection to the server. Clients can publish or subscribe to messages, as well as unsubscribe or delete messages from applications, and can also disconnect from the server. The MQTT server, called the broker, can be an application or a device which between publishers and subscribers. It can accept messages and process related requests.

2.2. MQTT Characteristics

The currently available MQTT protocol is in the version MQTTv3.1.1 released in 2014, aiming to run in networks such as TCP or IP.MQTT' s publish-subscribe model allows for seamless bi-directional communication between devices, and in this model, the client that sends a message (the publisher) is decoupled from the client or clients that receive the messages (or the subscribers), which makes it much easier to decouple from the applications. Moreover, it has a compact header, which only contains 2 bytes, and simple protocol interaction, that minimizes data transmission and protocol exchange, to reduce network traffic. In the case of abnormal interruption, it has a notification mechanism to notify the publisher, combined with Last Will and Testament functions to inform the relevant clients of abnormal interruption.

MQTT has 3 defined QoS (Quality of Service) levels, making it convenient to choose flexibly according to different scenario requirements. Facilitate application according to different scenario requirements. The QoS is a key feature of the MQTT protocol that enables clients to select service levels matching with their network reliability and application logic. Since the MQTT protocol guarantees transmission, even though when the underlying transmission is unreliable, the QoS makes it easier to communicate in unreliable networks.

QoS 0 (up to once): the client publishes the message only once, regardless of whether the message has reached the broker. QoS 1 (at least once): the client publishes the message repetitively until the broker responds to confirm the receipt, which may cause the message to reach the broker multiple times. QoS 2 (exactly once): the client publishes the message once while ensuring that it reaches the broker, and the QoS 2 communication requires higher bandwidth than the QoS 0 and QoS 1.

In the MQTT protocol, one session can contain multiple subscriptions. Each subscription in each session will have a different topic filter. When every client has connected with the server, a state of

 doi:10.20944/preprints202401.1250.v1

interaction will occur. Session may only last for one network connection, or if the client can re-establish the connection before the session expires, the session may exist across multiple network connections. When the identifier can be matched with the server connected to, the server will publish messages to all clients that have the matching identifier.

3. Materials and Methods

In the past IoT transmission technology, only a single communication protocol technology was used to realize the data transmission. In this paper, data is transmitted based on MQTT protocol and OneNET Cloud Platform. The scheme utilizes the OneNET Cloud Platform, which leverages the characteristics of MQTT protocol technology and enhances data transmission quality. It facilitates later management and ensures data security. All the above are completely different from the traditional IoT communication technology. The overall structure of the scheme is illustrated in Figure 3.

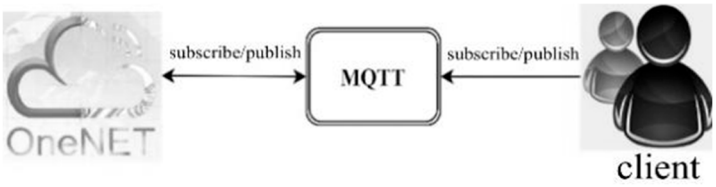

Figure 3. overall structure of the scheme.

The hardware part is composed of embedded devices which can be connected to the OneNET Cloud Platform through the MQTT protocol by using AT instructions. AT is the short form of Attention. Each AT command line can contain a single AT instruction. Only one AT instruction can be included per AT command line. The transmission of AT instructions, in addition to "AT" these two characters, can receive up to 1056 characters long (including the last blank character). For the URC instructions or responses actively reported to the PC by the terminal devices, there must also maximize one AT instruction in each AT command line, against multiple instructions or responses in the reported line. AT command terminates with the carriage return, and the response or report process is terminated with a carriage return wrap. Finally, there will be certain feedback on whether each command is executed or completed.

The purpose of applying AT instructions is to simplify the complexity of networking embedded devices. The command is short, easy to understand and can basically complete all the functions of wireless communication technologies (call, SMS, fax, etc.), additionally provides a set of standard hardware interface, and the serial port simplifies the hardware design. Common format of the AT command: AT + CMD = <xxx> [, <xxx>, <xxx>].

4. Results

4.1. MQTT Communication Debugging

Discover the product and service series corresponding to the device through the MQTT protocol, and then connect to the China Mobile OneNET Cloud Platform. After successful connection, the cloud platform would display the information of the device currently used, such as the name of the device manufacturer, device model, etc., and could receive data and send commands. The cloud platform has rich interfaces, which can facilitate communication and upgrade function in the product and service series corresponding to the device through the MQTT protocol, and then connect to the China Mobile OneNET Cloud Platform. After successful connection, the cloud platform would display the information of the device currently used, such as the name of the device manufacturer, device model, etc., and could receive data and send commands. The cloud platform has rich interfaces, which can facilitate communication and upgrade function.

Research manuscripts reporting large datasets that are deposited in a publicly available database should specify where the data have been deposited and provide the relevant accession numbers. If the accession numbers have not yet been obtained at the time of submission, please state that they

 doi:10.20944/preprints202401.1250.v1

will be provided during review. They must be provided prior to publication. Intervention studies involving animals or humans, and other studies that require ethical approval, must list the authority that provided approval and the corresponding ethical approval code.

When entering the OneNet Cloud Platform for the first time, it is necessary to create an account, log in, find the corresponding product services, then create the products, add the devices, and then the devices can be connected to the OneNet Cloud Platform. For ease of testing, the software MQTT.fx can be used to simulate the connection to the OneNet Cloud Platform for debugging. What's more, it is also necessary to input the ID of the product, the device name and key. And then, convert the device key to the token by the token password generation tool is also in need. For each device has different keys, it means login passwords are in differ, thus the security is improved.

The password after conversion is the login password on the software MQTT.fx. The IP address of MQTT is 183.230.40.96 and port number is 1883. After successful login, click Connect, and when the prompt message turns green, it means the login is successful. At this time, the devices can be seen online on the OneNet Cloud Platform, and the devices will be supervised online.

Next, subscribe to the topic and publish the message on the MQTT.fx software side. The format of the published message must be in JSON format. After that, the published messages would be seen on the OneNet cloud platform. The software side shows that the message has been published and subscribed to with a quality-of-service level of QoS0. Use different levels to get messages reach the destination on demand, adapted to the needs of unstable network transmission. OneNet The uploaded real-time data messages can be seen on the cloud platform, achieving the function of receiving data messages in real time.

The cloud platform can also monitor each different message type separately, and record the message upload status of each time, which is easy for data management in the later stage. This part, observing the update and change of data information in real time based on big data visualization analysis[13-15], is shown in Figures 4 and 5.

Figure 4. valve opening Information.

Figure 5. Motor Information.

 doi:10.20944/preprints202401.1250.v1

The performance of the MQTT protocol compared with that of HTTP IoT and the TCP message overhead are shown in Table 1.

Table 1. TCP consumption.

	MQTT byte	HTTP byte
Connection establishment	5 572	2 261
Break	376 (Optional)	0
Every message posted	388	3 285
The sum of one message	6 336	5 546
The sum of 10 messages	9 829	55 460
The sum of 100 messages	44 748	554 600

It could be found that the connection establishment number for MQTT is larger because the MQTT CONNECT packets are exchanged after the TCP connection is completed. The equivalent "overhead" of HTTP must be included in every request, which is the reason why the overhead per message is much bigger. The MQTT disconnect overhead shown here can be 0, that is because it is not required until the TCP connection is closed. Table 2. has tabulated the comparison of response times.

Table 2. Response time.

Average MQTT response time per message (ms) (QoS 1)	Average HTTP response time per message (ms)
113	289
47	289
43	289

Since more information were sent in a single TCP connection, it would be feasible to reduce the response time, which is because there was no extra cost that would be used to create connections for each message. HTTP does not have such capabilities and some methods, such as caching or connection pooling, can optimize the response speed and overhead, but not to the MQTT level.

These measurements were adapted to MQTT QoS 1, waiting for a response one at a time before the next message. Before receiving the above responses, performance is improved by transmitting subsequent messages (at the cost of some coding complexity), or by adopting QoS 0 not requiring a response (at the cost of some reliability).

It would be possible to gain the additional relative throughput and response time with QoS 0 and QoS 1, the following are the numbers obtained through the local HiveMQ broker using the round-trip scheme, as shown in Table 3.

Table 3. Comparison of service quality.

MQTT QoS	Number of messages per second	Average round-trip time (milliseconds)
1	1 234	0.81
0	18 416	0.054

Compared separately the MQTT's total traffic consumption is under 1 hour with HTTP's, and assuming a 5-min active heartbeat interval. This is shown in Figure 6.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 January 2024 doi:10.20944/preprints202401.1250.v1

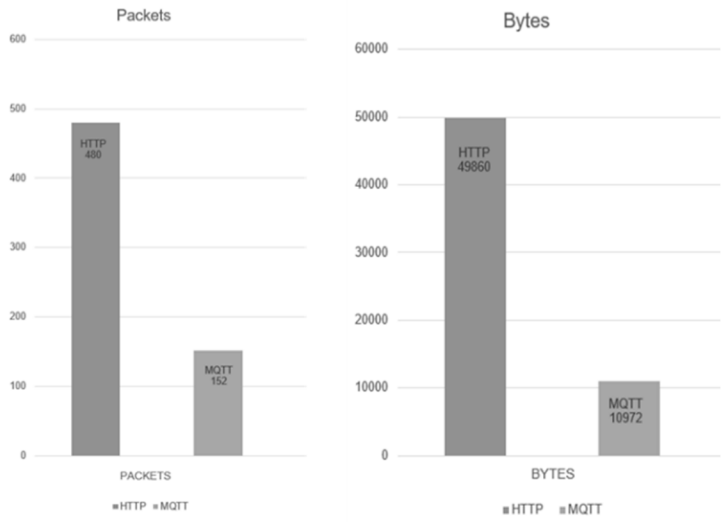

Figure 6. Total traffic consumption in one hour.

4.2. Equipment Debugging

The Embedded module device uses the AT+MQTTCFG command directly to create an MQTT connection, and the product ID, device name and key should be known when using this command. The command AT+MOTTOPEN is to start the MQTT server, and the command AT+MQTTSUB is used to subscribe to a topic, and only after subscribing to the topic successfully, it will publish. The module publishes topics, topics, and messages using the AT + MQTTPUB command, resets the module using the AT+CMRB command, using the AT + CSQ command to query the signal, and using AT + CGPADDR = 1 to check whether the IP is connected (return + CGPADDR: and the IP is connected). The whole process is shown in Figure 8.

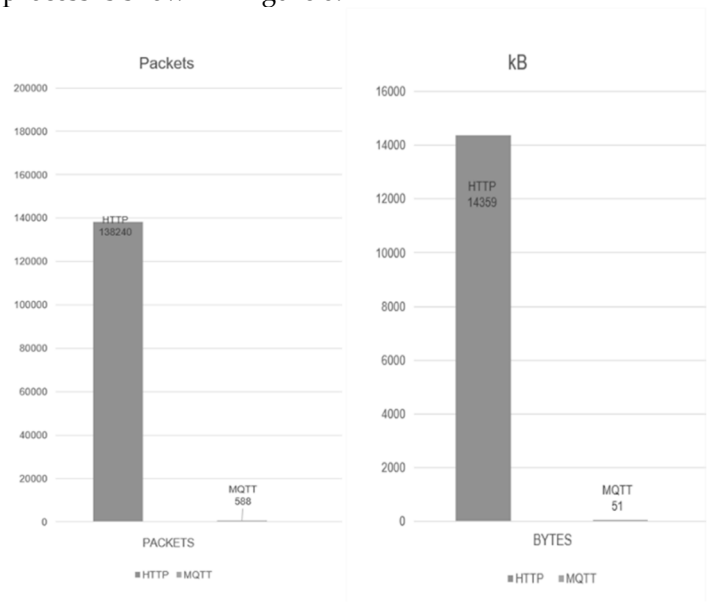

Figure 7. Total traffic consumption in one day.

 doi:10.20944/preprints202401.1250.v1


```
AT+MQTTSUB="$sys/512222/M5311/dp/post/json/+",0
OK

+MQTTSUBACK: 1,0,$sys/512222/M5311/dp/post/json/+
AT+MQTTSUB="$sys/512222/M5311/dp/post/json/accepted",0
OK

+MQTTSUBACK: 2,0,$sys/512222/M5311/dp/post/json/accepted
AT+MQTTSUB="$sys/512222/M5311/dp/post/json/rejected",0
OK

+MQTTSUBACK: 3,0,$sys/512222/M5311/dp/post/json/rejected
AT+MQTTSUB="$sys/512222/M5311/cmd/request/+",0
OK

+MQTTSUBACK: 4,0,$sys/512222/M5311/cmd/request/+
AT+MQTTSUB="$sys/512222/M5311/cmd/response/+/+",0
OK

+MQTTSUBACK: 5,0,$sys/512222/M5311/cmd/response/+/+

AT+MQTTPUB="$sys/512222/M5311/dp/post/json",1,0,0,64,"7b226964223a3130302c226470223a7b2276616c7665
6f70656e696e67223a5b7b2276223a33392c7d5d2c226d6f746f72223a5b7b2276223a34342c7d5d7d7d"
OK

+MQTTPUBACK: 28,0
+MQTTPUBLISH: 0,0,0,0,$sys/512222/M5311/dp/post/json/accepted,10,{"id":100}
```

Figure 8. Connect to the MQTT server.

Connect the module devices to the OneNet Cloud Platform, open the intelligent module serial port assistant of the IoT to connect to the devices, and return some basic information, indicating that the serial port connection is successful. Send the corresponding AT instructions, and the device is connected to the OneNet Cloud Platform, as shown in Figure 9.

```
AT+MQTTPUB="$sys/512222/M5311/cmd/response/0509ac97-6252-4334-9bce-f00ca1d0d077",1,0,0,0,"ValvePos"
OK

+MQTTPUBACK: 24,0
+MQTTPUBLISH: 0,0,0,0,$sys/512222/M5311/cmd/response/0509ac97-6252-4334-9bce-f00ca1d0d077/accepted,0,
```

Figure 9. Devices are connected to the cloud platform.

After the module devices are connected, it is turned to subscribe to the topic, publish the topic and message. Using the AT + MQTTPUB command, the valve information can be sent to the cloud platform, and the cloud platform can make a corresponding reply after receiving the corresponding information. The whole process is shown in Figure 10.

```
AT+SM=LOCK
OK
AT+CPIN?
+CPIN: READY

OK
AT+CGACT?
+CGACT: 1,1

OK
AT+MQTTCFG="183.230.40.96",1883,"M5311",60,"512222","version=2018-10-31&res=products%2F512222%
2Fdevices%2FM5311&et=1735660800&method=md5&sign=PaGbbKBS0AD75DoWly0Cow%3D%3D",1
OK
AT+MQTTOPEN=1,1,0,0,0,"",""
OK

+MQTTOPEN: OK
AT+MQTTSTAT?
+MQTTSTAT: 5
```

Figure 10. Subscription and Publishing.

Deliver the ValvePos command from the cloud platform, set the valve action and delay time to 30s. During the delay time, the module terminal needs to complete the data receiving, and perform

 doi:10.20944/preprints202401.1250.v1

corresponding operations on the valve, as well as provide information feedback to the cloud platform. The process is shown in Figure 11.

Figure 11. The module equipment reply.

Finally, the MQTT protocol is used to achieve the information distribution effect of one-to-many and two-way transmission. Combined with the use of the OneNet Cloud Platform, it is much easier to manage data, improves data transmission efficiency, and reduces costs. The Publish-Subscribe model simplifies application development and facilitates the transmission of messages. What's more, the QoS support ensures reliable transmission and improves message service quality.

5. Discussion

A significant revelation of our research is the pressing need for enhanced network bandwidth and data processing capabilities. In the big data era, the sheer volume of data produced by a myriad of IoT devices necessitates a system capable of handling this influx efficiently. This requirement extends beyond mere data handling; it calls for scalable and maintainable IoT infrastructure that can adapt to evolving technological demands. Addressing the compatibility issue is also paramount. To achieve a certain degree of specialization, it is often necessary to utilize equipment from various manufacturers, each choosing their own technical standards, operating systems, and protocols. This diversity in technology and specialization can create conflicts, posing a formidable challenge in ensuring seamless interoperability among different IoT components. The need to harmonize these diverse elements is critical for the cohesive functioning of IoT systems.

Furthermore, our study highlighted a noteworthy concern regarding MQTT - the potential escalation in bandwidth consumption and latency issues, especially at higher quality levels. This observation is critical for IoT applications where real-time data transmission and processing are paramount. Looking ahead, several avenues for future research have become evident. First, addressing the bandwidth and message capacity constraints is crucial. This can be achieved by developing more robust infrastructure capable of handling increased data loads without compromising on speed or efficiency.

Enhancing the security framework for MQTT is another critical area. Implementing stringent encryption protocols and complex authentication mechanisms can significantly bolster the security aspect, which is paramount in the IoT domain. Additionally, broadening the support for diverse application scenarios and pushing towards more standardized practices in IoT communications will not only streamline operations but also foster a more cohesive IoT ecosystem.

6. Conclusions

This paper introduces a groundbreaking integration of the MQTT protocol with the OneNet Cloud Platform within a big data environment, significantly enhancing IoT communication technologies. Our study delved into the intricacies of IoT communications facilitated by this integration, demonstrating how it ensures high-speed and stable data exchange, and greatly simplifies the management of the voluminous data generated by IoT devices. Importantly, this approach also establishes robust security measures, addressing a critical concern in today's IoT applications.

These advancements offer considerable benefits for IoT applications, particularly in environments requiring reliable and efficient communication. However, our findings also highlight certain areas that need further attention, underscoring the robust potential of this integration while pointing out its limitations. For instance, while we have made strides in data management and security, there remain challenges in ensuring the sustainability and relevance of IoT systems in the rapidly evolving technological landscape.

In conclusion, the integration of MQTT with the OneNet Cloud Platform in a big data environment presents a promising frontier for IoT communications. Future work should focus on addressing the identified limitations, exploring further optimizations, and investigating potential

applications in diverse IoT domains. By doing so, we can enhance not only the efficiency and security of IoT systems but also ensure their long-term viability and adaptability to future technological advancements.

Author Contributions: Conceptualization, Youcheng Shan. and Yuening Su.; methodology, Youcheng Shan and Yuening Su.; software, Yuening Su and Jiahui Lin.; validation, Youcheng Shan, and Tingyu Shan.; formal analysis, Jiahui Lin.; investigation, Tingyu Shan.; resources, Youcheng Shan.; data curation, Yuening Su.; writing-original draft preparation, Yuening Su.; writing-review and editing, Jiahui Lin and Tingyu Shan.; visualization, Youcheng Shan and Tingyu Shan.; supervision, Youcheng Shan.; project administration, Youcheng Shan.; funding acquisition, Youcheng Shan and Tingyu Shan. All authors have read and agreed to the published version of the manuscript." Please turn to the CRediT taxonomy for the term explanation. Authorship must be limited to those who have contributed substantially to the work reported.

Funding: This work was supported by 2022 National University and Vocational College Logistics Education Reform and Research Project of China Logistics Association (JZW2022006).

Conflicts of Interest: The author declares that there is no conflict of interest regarding the publication of this paper.

References

1. Phuc B H, Van Q P, Linh N Q, et al. "Dynamic Threading to Improve Embedded Software Performance in IoT Devices Using MQTT Protocol", 2018 International Conference on Advanced Technologies for Communications (ATC), Ho Chi Minh City, Vietnam, 2018, 321-325.
2. SHAN Y C. "Research on the Innovation Path of Cultural and Art Financial Driven by Big Data", International Aid, 2020,40(9): 126-127.
3. SHAN Y C. "Social Network Text Sentiment Analysis Method Based on CNN-BiGRU in Big Data Environment". Mobile Information Systems, 2023,2023(1): 1-8.
4. SHAN Y C. "Research on Integer programming Strategy of 5G Base Station Location Based on Big Data", Communication Technology, 2023,56(10): 1166-1172.
5. Chen M, Hua Q J, Gu X H, et al. "Design and development of water data transmission system based on MQTT", Industrial Control Computer, 2022,35 (05): 6-8.
6. Xie X Z, Chen X G. "MQTT Protocol Access to OneNET IoT Application", *Woodworking Machine Tool*, 2021 (02): 10-13,20.
7. Zeng Z R, Chen D J, Xiao Y. "IoT Communication Protocol based on Blockchain and MQTT", *Electronic Technology*, 2022,51 (05): 15-19.
8. Shi Y H, Lu Y, Zhang J. "Design and Implementation of Equipment Cabinet Management System based on MQTT", *Industrial Control Computer*, 2021,34 (03): 108-109,111.
9. He Z Y, Ma Y H. "Design of Smart Home Security System based on MQTT", *Residential and Real Estate*, 2020, 30(10): 59-60.
10. Chen W Y, Gao J, Yang H. "Design and Implementation of IoT Communication System based on MQTT Protocol", *Journal of Xi'an University of Posts and Telecommunications*, 2020,25 (03): 26-32.
11. Wang R, Wu Y J, Li Y P. "Design of IoT Practical Training Cloud Platform based on MQTT Protocol", *Industrial Control Computer*, 2018,31 (09): 101-103.
12. Long Q L, Niu D X, Lin L Y. "Smart Energy-saving Control System based on OneNET Cloud Platform and IoT MQTT Protocol", *Computer Measurement and Control*, 2021,29 (07):127-130, 135.
13. Qiu K. "Implementation and Application of Big Data Visualization Analysis based on Hadoop Platform", *Electronic Technology and Software Engineering*, 2022,237(19):184-187.
14. Yan Z. "Research on Visualization Analysis of Academic Situation Data Based on Density Clustering Algorithm in Electronic Technology and Software Engineering", *Modern Computers*, 2022,28(6):43-47.
15. Chen L B, He X Y, Yao H. "Research History and Prospects of Historical Architecture in China - Based on CiteSpace Data Visualization Analysis", *Urban Architecture*, 2022,19(1):122-126.