Journal of Electronic Research and Application, 2024, Volume 8, Issue 5
http://ojs.bbwpublisher.com/index.php/JERA
ISSN Online: 2208-3510
ISSN Print: 2208-3502

The Design of Multi-Connector IoT Central Gateway Based on Raspberry Pi and Docker

Jie Wang*

Sichuan Vocational and Technical College, Suining 629000, China

***Corresponding author:** Jie Wang, cafebabe1314@gmail.com

Copyright: © 2024 Author(s). This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY 4.0), permitting distribution and reproduction in any medium, provided the original work is cited.

Abstract: With the vigorous development of Internet of Things (IoT) technology, the demand for communication and data exchange between different types of IoT devices is increasing day by day. To solve the problems of diversity and complexity of communication protocols between devices, this paper proposes a design scheme of a multi-connector IoT central gateway based on Raspberry Pi and Docker. Through the research and application of related technologies, by integrating multiple communication interfaces and utilizing containerization technology, an efficient, flexible, and scalable IoT central gateway has been realized, which can support the connection and data interaction of multiple communication protocols and provide strong support for the stable operation and development of the IoT system.

Keywords: Internet of Things; Raspberry Pi; Docker; Connector; Central Gateway

Online publication: October 8, 2024

1. Introduction

With the rapid development of Internet of Things (IoT) technology, an increasing number of devices are connected to the network, forming a huge and complex IoT system. However, different IoT devices often adopt different communication protocols and interfaces, which brings huge challenges to the interconnection and data exchange between devices. To solve this problem, the IoT central gateway emerges as the times require. As a bridge connecting different devices and networks, it is responsible for important tasks such as protocol conversion, data processing, and transmission.

Among numerous technical solutions, Raspberry Pi has become an ideal choice for building an Internet of Things gateway due to its low cost, high performance, and rich interface resources. Meanwhile, the emergence of Docker technology has brought great convenience to the deployment and management of applications, improving the scalability and stability of the system. The design of the multi-connector Internet of Things central gateway based on Raspberry Pi and Docker proposed in this paper aims to achieve the integrated management of various Internet of Things devices by combining the hardware advantages of Raspberry Pi and the software containerization technology of Docker. We will elaborate on the architecture, key technology selection, implementation details, and performance evaluation results of this design.

2. Relevant technologies

2.1. Raspberry Pi

The Raspberry Pi is a small single-board computer based on the Linux operating system, developed by the Raspberry Pi Foundation in the United Kingdom (UK) [1]. Its design goal is to promote computer science education and technological innovation, especially to popularize computer programming and hardware development skills in the education field. It has various interfaces, such as Universal Serial Bus (USB), Ethernet, General-Purpose Input/Output (GPIO), and many more which can conveniently connect various sensors and devices. The emergence of the Raspberry Pi provides a low-cost and highly flexible platform for the development of Internet of Things applications.

2.2. Docker

Docker is an open-source containerization technology that allows developers to package applications and their dependencies into a portable container [2]. Docker containers have the advantages of being lightweight, having good isolation, and being quickly deployed, which can greatly improve the development, operation, and maintenance efficiency of applications. In the field of the Internet of Things, Docker can be used for the rapid deployment and management of various services and applications, achieving the flexible expansion of the system.

2.3. Internet of Things communication protocol

There are various communication protocols in the Internet of Things, such as Wi-Fi, Bluetooth, Zigbee, LoRa, among other things [3]. These protocols have their own characteristics and are suitable for different scenarios and application requirements. Wi-Fi has the advantages of high speed and long-distance transmission and is suitable for devices with high bandwidth requirements. Bluetooth is suitable for short-distance and low-power device connections. Zigbee has characteristics such as low power consumption and self-organizing network and is suitable for large-scale sensor networks. LoRa performs well in long-distance and low-power communication.

3. Overall system design

The IoT central gateway receives a large amount of data from numerous IoT devices in the entire IoT system and conducts aggregation and organization. This system uses Raspberry Pi as the hardware support for the IoT central gateway and installs and runs Docker containerization technology in Raspberry Pi. Various IoT devices need to be connected to the IoT central gateway. All connected IoT devices can be classified as serial port protocol devices and network protocol devices. The connection method for serial port protocol devices adopts direct physical serial port connection and network serial port connection, Transmission Control Protocol (TCP) Client, while network protocol devices are connected to the IoT central gateway according to different network protocols.

Various IoT devices have different access methods and different access protocols. To solve the problem of providing support for IoT devices with different protocols and different access methods in the IoT central gateway, connector development is carried out for the images in Docker for the IoT devices that need to be accessed, so that each IoT device that needs to be accessed has a connector template corresponding to it. When an IoT device needs to be connected to the IoT central gateway, select the connector template corresponding to the current IoT device in the connector template to create the connector.

In the IoT central gateway, a web page is provided to configure the connectors for the IoT devices that need to be accessed and to provide data visualization and actuator control functions for the already accessed IoT devices [5]. The IoT central gateway provides a TCP Server, Message Queuing Telemetry Transport Client (MQTT) Server data access protocols, and supports TCP Client, MQTT Client, and ThingsBoard (TB) Client to connect to

different types of cloud platforms[12–14]. Refer to **Figure 1** for the overall structure of the IoT center gateway.

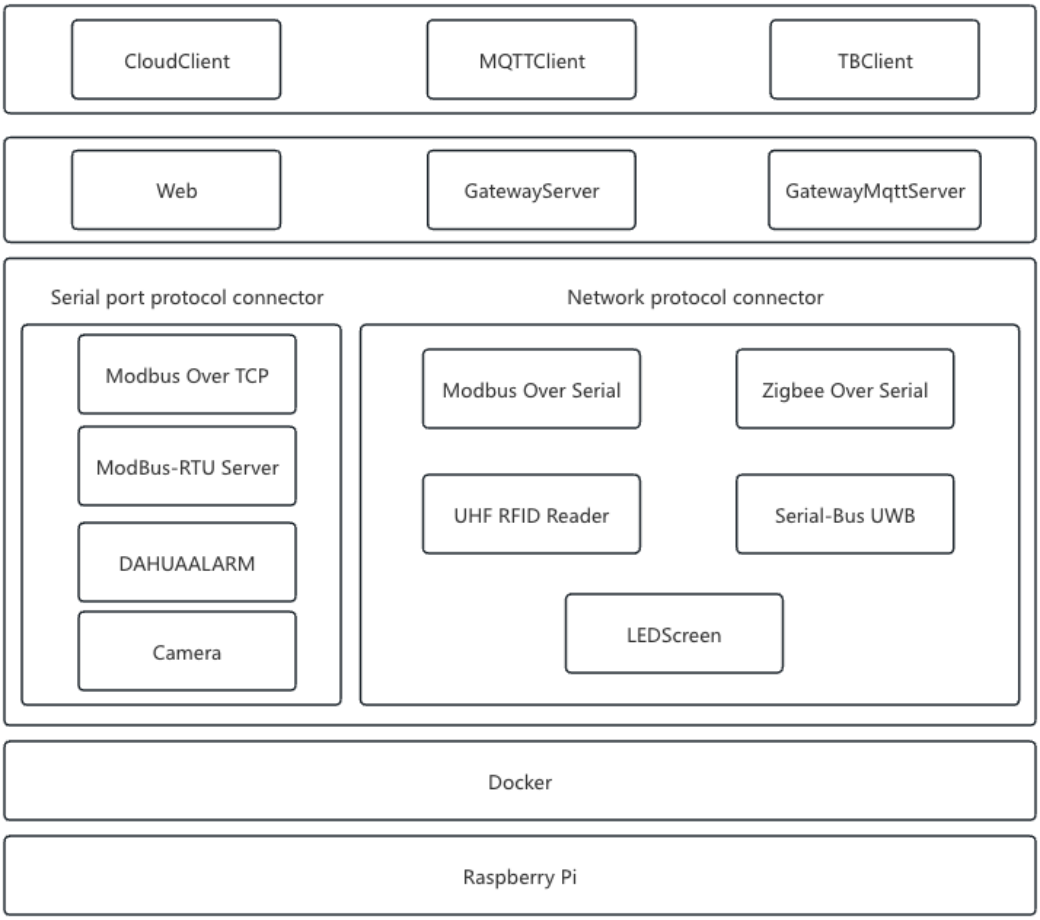

Figure 1. Overall structure diagram of the Internet of Things central gateway

4. System design and implementation

4.1. Main modules of the system

The Raspberry Pi is selected as the core motherboard, which has strong performance and rich interface resources. Docker is installed and deployed in the Raspberry Pi. The implementation of each module uses the corresponding Docker image. For example, Node-RED is used to implement the message processing module, and the interface code with other modules is written. The security authentication module uses the Open Authorization 2 (OAuth2) protocol and Transport Layer Security (TLS) encryption technology to ensure the secure access and communication of devices and users. The images in Docker are regarded as the connector templates in the IoT central gateway, and the containers in Docker are regarded as the connectors in the IoT central gateway. We develop some common connector templates for different types and protocols of IoT devices for users to use.

When an IoT device needs to be connected to the IoT central gateway, the central gateway provides a web page with the function of creating a connector for the IoT device to be connected. When this device is connected to the IoT central gateway, the function of viewing and controlling sensor data is provided in the gateway. The IoT central gateway supports different protocols to connect to the IoT cloud platform.

4.2. Front-end design

The front-end development framework adopts Vue.js [6]. Vue.js uses a reactive data binding mechanism, which

keeps the view and data synchronized. When the data changes, the view will be updated automatically. It is a concise, flexible, and easy-to-use front-end framework, suitable for building web applications of various scales. In the web application, functions such as connector creation and IoT device data viewing are provided. Administrators conduct remote management and monitoring of the IoT gateway.

4.3. Back-end design

The web back-end application program is developed using the Flask framework, providing RESTful API for the background data access service [7,8]. Through these interfaces, administrators can view the device list, monitor the data flow, and perform firmware upgrades and other operations, thereby realizing the real-time management and response of the Internet of Things environment.

4.4. Protocol conversion

(1) Define a unified data format: Design a universal data format, including fields such as source address, destination address, data type, and data content, as the intermediate format for conversion between different protocols [9].

(2) Protocol parsing and conversion: For each communication protocol, write the corresponding parsing program to parse the received data into the unified format, and perform conversion and encapsulation according to the target protocol.

5. System evaluation and performance analysis

To evaluate the effect of the designed IoT central gateway in practical applications, we conducted a series of performance tests and analyses. They mainly include tests on the response speed, resource occupation, and security performance of the system.

5.1. Response speed

By simulating the scenario of multiple devices concurrently connecting, the response speed and processing capacity of the gateway under different loads were tested. The results showed that the designed gateway was able to handle a large number of concurrent requests while maintaining stability, and the response time met the expected requirements.

5.2. Resource occupation

Monitor the Central Processing Unit (CPU) utilization, memory consumption, and storage space usage of Raspberry Pi. Due to the lightweight feature of Docker, the system performs well in terms of resource consumption and can run multiple modules under a lower hardware configuration and maintain stability.

5.3. Security performance

The security performance of the IoT central gateway was tested through network packet capture and vulnerability scanning tools. We found that data transmission encrypted by TLS and OAuth2 authentication ensured the security and integrity of device and user data. In addition, the security configuration and access control functions of NGINX effectively prevented potential network attacks and malicious access.

5.4. Practical application cases

To verify the feasibility and effectiveness of the designed IoT central gateway in practical applications, we

conducted deployment and testing in the environment of a smart home company. In this application, the gateway successfully connected and managed various devices, including temperature sensors, smart sockets, security cameras, and many more. Administrators monitored and controlled the devices in real-time through the remote management interface, and the system stability and performance were satisfactory.

6. Discussion and outlook

The multi-connector Internet of Things central gateway based on Raspberry Pi and Docker designed in this paper has made significant progress in performance and scalability. However, there are still some challenges and room for improvement, to further optimize the system response speed and resource consumption, especially the performance when handling large-scale device connections. Promote the standardization of communication protocols and data formats of Internet of Things devices to enhance the interoperability and integration between devices.

Future research can further explore how to utilize edge computing and artificial intelligence technologies to enhance the ability of the Internet of Things central gateway in data processing and decision support. Additionally, with the popularity of 5G technology and the expansion of Internet of Things application scenarios, how to achieve higher-speed and lower-latency data transmission is also an important research direction.

7. Conclusion

This paper presents the design of a multi-connector Internet of Things central gateway based on Raspberry Pi and Docker, effectively demonstrating how to utilize open-source hardware and containerization technology to build a cost-effective and flexible Internet of Things solution. Through the verification of system design and practical applications, we have proved the superior performance and feasibility of this gateway in managing and connecting various IoT devices. Future work will continue to explore and optimize this design to meet the growing demands of Internet of Things applications.

Disclosure statement

The author declares no conflict of interest.

References

[1] Liu L, Wang X, Wang G, et al., 2024, Design of Intelligent Gateway Based on Raspberry Pi and MQTT. Mechanical and Electrical Engineering Technology, 2024(04): 1–6.

[2] Gong R, Gong Y, Li Z, et al., 2024, Management System of Distributed Epidemic Data Visualization Platform Based on Docker. Shanxi Electronic Technology, 2024(03): 84–87.

[3] Zeng D, 2022, Design and Implementation of Communication Transmission in the Intelligent Water Conservancy Perception System. Digital Communication World, 2022(11): 32–34 + 94.

[4] Guo S, 2020, Design and Implementation of Distributed General Internet of Things Data Management Platform, thesis, Beijing University of Posts and Telecommunications.

[5] Xu Y, 2024, Design of Internet of Things System Based on Embedded Web Intelligent Monitoring. Application of Integrated Circuits, 41(05): 286–287.

[6] Deng J, Tong B, Han J, et al., 2024, Design and Implementation of RH Hydraulic System in Refining Furnace Based

on MQTT Protocol and Vue.js Framework. Industrial Control Computer, 37(03): 33–35.

[7] Chen R, 2024, Application of Python and Flask in Enterprise WeChat Message Reception. Fujian Computer, 40(04): 53–56.

[8] Zhang H, Cheng X, Xu H, 2022, Design and Implementation of RESTful API Interface for IoT Elevator Data Interaction Platform. Digital Technology and Application, 40(10): 171–174.

[9] Tan C, Xu H, Fu Q, 2024, Design of Multi-protocol Data Conversion Gateway for Industrial Sensors. Instrument Technique and Sensor, 2024(01): 70–75.

[10] Li Z, Wang W, Wang K, 2023, Design and Implementation of Portal System Based on OAuth2.0 Protocol. China Digital Medicine, 18(12): 47–51.

[11] Jin H, Yu A, 2023, Practical Application of Nginx Load Balancer Cluster Architecture. Journal of Anhui Vocational College of Electronics & Information Technology, 22(03): 1–6.

[12] Qiu Y, Jing B, Li J, 2019, Design of Internet of Things Sensor Node Based on ESP8266 WiFi Module and MQTT Protocol. Internet of Things Technology, 9(6): 24–26.

[13] Leng M, 2024, Remote Experiment Monitoring System Based on LabVIEW and TCP. Industrial Control Computer, 37(06): 15–17.

[14] Liu S, 2023, Design of Portable Internet of Things Meteorological Station System Based on ThingsBoard Platform. Information and Computer (Theoretical Edition), 35(24): 108–110.

[15] Chen X, Zhang X, Zhang J, et al., 2024, Design and Implementation of Cloud Intelligent Production Line Control System Based on Node-RED. Shandong Industrial Technology, 2024(03): 58–62.

Publisher's note

Bio-Byword Scientific Publishing remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.