SAKARYA UNIVERSITY JOURNAL OF COMPUTER AND INFORMATION SCIENCES

http://saucis.sakarya.edu.tr/

e-ISSN: 2636-8129
Publisher: Sakarya University

Vol. 7, No. 3, 494-509, 2024
DOI: 10.35377/saucis...1504759

RESEARCH ARTICLE

MQTT in Action: Building Reliable and Scalable Home Automation Systems

Maysaa Salama[1, *], Bilal Raslen [1]

[1] Faculty of Computer and Information Sciences, Computer Engineering, Sakarya University, Türkiye

Corresponding author:
Maysaa SALAMA,
Sakarya University, Faculty of Computer
and Information Sciences, Computer Engineering
Türkiye
maysaa.salama@ogr.sakarya.edu.tr

ABSTRACT

This study presents the development and implementation of an MQTT-based home automation system, emphasizing its reliability, scalability, and efficiency. The system integrates the ESP8266 microcontroller with various sensors, including IR, temperature, DHT11, sound, and vibration sensors. Leveraging the lightweight MQTT protocol for real-time data transmission and PostgreSQL for robust data storage, the system monitors environmental conditions, detects specific events, and performs appropriate actions. Performance tests indicate the system maintains low response times, high data accuracy, and exceptional reliability, making it a viable solution for modern smart homes. The findings suggest that the system enhances security, optimizes energy usage, and improves overall comfort for homeowners. Future research will focus on enhancing the security of the MQTT protocol against spoofing attacks through the integration of AI for threat detection and blockchain technology for enhanced data security.

Article History:
Received: 25.06.2024
Accepted: 26.12.2024
Published Online:31.12.2024

Keywords: MQTT, Home Automation, IoT, ESP8266, Sensors, Real-time Monitoring, Data Transmission, Reliability, Scalability, PostgreSQL

1. Introduction

Message Queuing Telemetry Transport (MQTT) is a machine-to-machine (M2M) Internet of Things connectivity protocol designed as an extremely lightweight publish/subscribe messaging transport. It is ideal for connecting remote devices with a small code footprint and minimal network bandwidth. MQTT is widely used across various applications, from mobile applications to lightweight communication in automotive and industrial systems, most notably, home automation [1]. In the realm of home automation, MQTT stands out for its simplicity and efficiency, which are crucial in environments where bandwidth and power usage are limited. Home automation systems need robust and scalable communication protocols to manage the many devices and frequent message exchanges typical in smart homes [2]. MQTT's capability to provide real-time updates and maintain stable connections, even over unreliable networks, makes it particularly suitable for this role. This functionality enables various applications, from basic home security to intricate automation scenarios. This paper explores the implementation of MQTT in home automation systems, emphasizing its reliability and scalability. The paper aims to analyze the advantages of MQTT over other communication protocols in terms of system reliability and data handling efficiency. It will demonstrate through case studies how MQTT can be implemented in home automation to achieve scalable and secure systems and will address the challenges faced during the implementation of MQTT in these systems, proposing viable solutions. In addition to its efficiency and scalability, securing MQTT communications is critical, especially in home automation systems where sensitive data is exchanged between devices. MQTT broker authentication and authorization mechanisms are essential for preventing unauthorized access. Typically, MQTT brokers support username and password authentication, where each client must provide valid credentials to establish a connection. To further strengthen security, OAuth or token-based authentication can be employed, which provides a more robust way to verify the identity of clients. Data encryption methods are also integral to maintaining the confidentiality and integrity of the data transmitted via MQTT. Transport Layer Security (TLS) is commonly used in conjunction with MQTT to encrypt data during transit, protecting it from eavesdropping or tampering. Using TLS, all communication between the client and broker is secured, ensuring that sensitive sensor data is protected. In addition to these measures, access control lists (ACLs) can be used to define specific permissions for each MQTT client, controlling what topics they can publish to or subscribe to, thereby reducing the risk of unauthorized actions. By combining these mechanisms, the system ensures a high level of protection against unauthorized access, data breaches, and potential cyber threats. With its designed efficiencies and growing adoption in various sectors,

Cite as: Salama M. and Raslen B.. “MQTT in Action: Building Reliable and Scalable Home Automation Systems”. *Sakarya University Journal of Computer and Information Sciences*. vol: 7 no: 3. pp: 494-509, 2024. Doi: 10.35377/saucis...1504759

MQTT is pivotal in advancing home automation systems [3]. It not only enhances how devices communicate but also ensures that these communications are reliable and scalable.

2. Related Work

The use of MQTT (Message Queuing Telemetry Transport) in home automation has gained significant attention due to its lightweight nature, efficiency, and scalability. This section reviews key advancements in MQTT applications, comparisons with alternative IoT protocols, security enhancements, and scalability improvements, all of which underscore MQTT's role in modern home automation systems.

Advancements in MQTT Applications

Many studies have explored the optimization of MQTT for smart home environments. For instance, in [4], the authors proposed a dynamic model for adjusting MQTT message frequencies to optimize power consumption and network efficiency, predicting network conditions to adjust message rates as needed [4]. This significantly reduced power usage while maintaining high system responsiveness. Another study examined MQTT's integration for managing energy distribution from renewable sources within smart homes, showcasing MQTT's role in promoting sustainable energy practices [5]. Moreover, it developed an MQTT-based framework that enhanced interoperability across various smart home devices and reduced latency, making it a powerful tool for unifying disparate devices into a cohesive network [6].

Comparative Studies with Other IoT Protocols

While MQTT is well-suited for low-bandwidth, reliable messaging, comparisons with other IoT protocols highlight its unique benefits and limitations [7] provided a rigorous comparison between MQTT and CoAP (Constrained Application Protocol), noting that MQTT offers superior throughput in high-demand scenarios. At the same time, CoAP performs more efficiently in energy-constrained settings. Another study by [8] contrasted MQTT with HTTP, emphasizing MQTT's enhanced security features and suitability for dynamic network environments typical of home automation. Additionally, [9] compared MQTT with AMQP (Advanced Message Queuing Protocol), finding that MQTT's lower overhead and support for real-time data transmission make it preferable for latency-sensitive applications.

In contrast to protocols like Zigbee and Z-Wave, which operate as mesh networks optimized for local device communication, MQTT's publish-subscribe architecture supports seamless device interoperability across broader network scales. Zigbee and Z-Wave excel in short-range, low-power operations within confined spaces, making them ideal for small-scale automation but limited in scalability and interoperability across different network infrastructures. Thread is an IP-based networking protocol for low-power mesh networks, while MQTT is a lightweight messaging protocol. MQTT can be used over Thread networks to provide efficient publish/subscribe messaging, making it advantageous in bandwidth-constrained environments. Another competing protocol, offers IP-based connectivity but lacks the lightweight simplicity of MQTT, which is advantageous in bandwidth-constrained environments. Overall, MQTT's flexibility and efficient message delivery make it a robust choice for IoT applications requiring both local and remote communication, whereas alternatives like Zigbee and Z-Wave may be better suited for exclusively local automation.

Security Enhancements in MQTT-Based Systems

Security remains a primary concern for MQTT-based home automation systems, which manage sensitive data and control critical home functions. Various studies have proposed robust security measures to counteract potential threats in MQTT networks. For instance, [10] introduced a security framework integrating advanced encryption standards and secure authentication to defend against cyber threats. Building on this, [11] developed a machine learning-based intrusion detection system (IDS) specifically for MQTT, which significantly improved real-time threat detection and mitigation. Additionally, [12] proposed a lightweight encryption scheme tailored for resource-constrained MQTT devices, enabling data protection without compromising performance.

Beyond spoofing attacks, MQTT-based systems are vulnerable to Denial of Service (DoS) attacks, Man-in-the-Middle (MitM) attacks, and data manipulation. DoS attacks can overload the MQTT broker with excessive requests, disrupting service availability. To mitigate this, rate-limiting mechanisms can be implemented at the broker level, alongside firewall rules and traffic monitoring to block malicious traffic early. MitM attacks, where an attacker intercepts and modifies communication, can be mitigated with Transport Layer Security (TLS 1.3), which encrypts data to maintain confidentiality and integrity. Enhanced authentication measures, such as token-based access and OAuth, ensure that only verified clients can access sensitive data streams. To detect data manipulation, digital signatures and message integrity checks verify message authenticity and content integrity. By combining these measures with IDS technology, which continuously monitors anomalous traffic, MQTT systems can provide a resilient security framework suitable for sensitive home automation applications.

Scalability and Interoperability Improvements

Scalability is essential for home automation systems that require expansion as new devices are integrated. Several studies have addressed scalability in MQTT-based systems. For example, [13] proposed a load-balancing technique that leverages MQTT to distribute workloads across multiple brokers, enhancing system reliability and fault tolerance. Additionally, it explored decentralized MQTT architecture, reducing the load on central brokers by enabling data processing at the network edge, which significantly improves scalability [14].

Interoperability is another important factor, particularly in multi-standard environments. Morgado-Dias and Quintal [15] examined MQTT's role as a bridging protocol for communication between heterogeneous devices, ensuring compatibility across different smart home ecosystems.

Furthermore, [16] explored MQTT’s integration with Zigbee and Z-Wave protocols, facilitating communication across diverse networks and enhancing system flexibility. These interoperability improvements make MQTT well-suited for smart home applications, as it can effectively integrate with other protocols and support a diverse array of devices.

Summary of Key Contributions

Table 1 summarizes the key contributions of MQTT applications for IoT systems, categorizing application advancements, comparative studies, security, and scalability.

Table 1 Key Contributions in MQTT Applications for IoT Systems

Contribution Area	Study	Key Findings
MQTT Applications	[4]	Optimized MQTT message frequencies for better power consumption and efficiency.
	[5]	Managed energy distribution from renewable sources in smart homes using MQTT.
	[6]	Improved interoperability and reduced latency with an MQTT-based framework.
Comparative Studies	[7]	MQTT vs. CoAP: MQTT has better throughput; CoAP is more energy-efficient.
	[8]	MQTT offers enhanced security compared to HTTP.
	[9]	MQTT has a lower overhead and is better for real-time applications than AMQP.
Security Enhancements	[10]	Developed a secure MQTT framework with advanced encryption.
	[11]	Machine learning-based intrusion detection for MQTT networks.
	[12]	Lightweight encryption for MQTT, enhancing security with low overhead.
Scalability and Interoperability	[13]	Load-balancing in MQTT for better reliability.
	[14]	Decentralized MQTT for improved scalability.
	[15]	Bridging protocol for seamless communication across devices.
	[16]	Integration of MQTT with Zigbee and Z-Wave for better interoperability.

3. System Architecture

The architecture of the MQTT-based home automation system is illustrated in *Figure 1*. This figure provides a visual representation of the interactions between various components, including sensors, the ESP8266 microcontroller, and the MQTT broker. The system is designed to monitor environmental conditions, detect specific events, and take appropriate actions based on sensor inputs.

Components and Workflow

The system consists of several interconnected components working seamlessly to enhance the smart home environment. The sound sensor acts as the initial trigger, detecting ambient sound levels and activating the IR sensor to monitor object movement. Upon detecting movement, the IR sensor signals the lighting system, prompting it to turn on the lights. This interaction ensures that the lighting is responsive to environmental changes.

Following movement detection, the IR sensor activates the temperature and humidity sensors, which measure ambient conditions to determine if adjustments to the indoor climate are necessary. Based on their readings, these sensors may reset the air conditioning system to optimize the indoor environment for comfort and energy efficiency.

Simultaneously, the vibration sensor operates continuously, monitoring for unusual movements or vibrations within the home. If the vibration sensor detects any anomalies, it activates the speaker (piezo) to alert the occupants, ensuring prompt awareness of potential security issues. This coordinated workflow among the sound, IR, temperature, humidity, and vibration sensors, along with the lighting and air conditioning systems, demonstrates the system's ability to maintain a secure, efficient, and comfortable home environment.

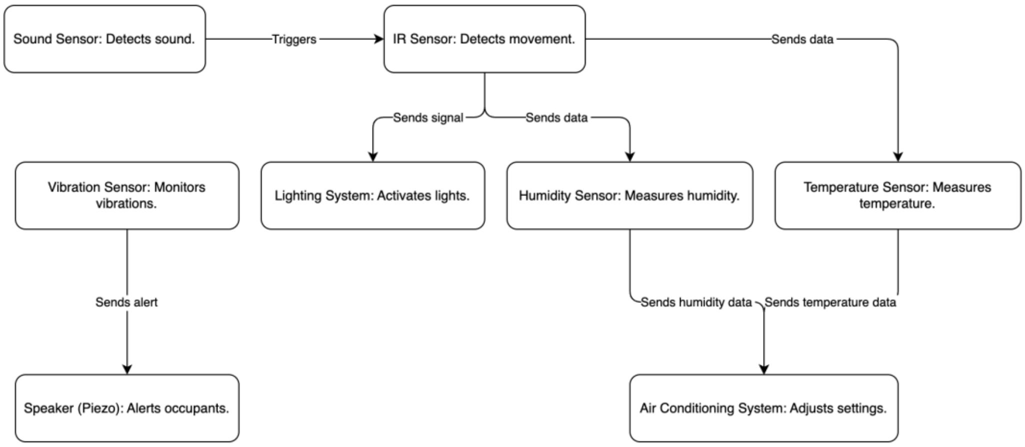

Sound Sensor: Detects sound.
Triggers
IR Sensor: Detects movement.
Sends data
Sends signal
Sends data
Vibration Sensor: Monitors vibrations.
Lighting System: Activates lights.
Humidity Sensor: Measures humidity.
Temperature Sensor: Measures temperature.
Sends alert
Sends humidity data
Sends temperature data
Speaker (Piezo): Alerts occupants.
Air Conditioning System: Adjusts settings.

Figure 1 System Architecture of the MQTT-based Home Automation System.

The ESP8266 microcontroller acts as the central hub for data collection and communication. It reads data from the connected sensors and transmits it to an MQTT broker using the MQTT protocol. The MQTT broker facilitates communication between the sensors and the data storage system.

As shown in *Figure 1* key interactions and data flows include:

- **Sensor to ESP8266:** Sensors continuously send data to the ESP8266 microcontroller.
- **ESP8266 to MQTT Broker:** ESP8266 formats the sensor data and transmits it to the MQTT broker.
- **MQTT Broker to Database:** The MQTT broker relays the data to a PostgreSQL database for storage and analysis.

The diagram showcases the interconnected nature of the system, highlighting the role of each component and the flow of information. This integrated approach ensures real-time monitoring, efficient communication, and effective management of the home automation system.

4. Methodology

This section delineates the comprehensive methodology employed in the development and implementation of the MQTT-based home automation system. The system architecture integrates the ESP8266 microcontroller, an array of sensors (IR, temperature, DHT11, sound, and vibration), the MQTT protocol for efficient communication, and PostgreSQL for robust data storage.

The experimental setup comprises multiple ESP8266 microcontrollers connected to various sensors to monitor environmental conditions and detect specific events. As shown in Figure 2, the setup includes the DHT11 temperature and humidity sensor, a sound sensor, an IR sensor, and a vibration sensor, all mounted on a breadboard. This configuration allows for efficient data collection and transmission for analysis.

We implemented robust authentication and authorization mechanisms within the MQTT broker to ensure the security of data transmitted within the system. Each ESP8266 microcontroller is required to authenticate using a username and password before it can connect to the MQTT broker. For enhanced security, we have also integrated TLS (Transport Layer Security) encryption, which ensures that all data transmitted between the sensors, broker, and database is encrypted, safeguarding it from potential eavesdropping or interception. Furthermore, access control lists (ACLs) are configured on the MQTT broker to restrict each sensor's access to specific topics, preventing unauthorized publishing or subscribing. By utilizing TLS and ACLs, we guarantee that only authorized devices can access the network, and the integrity of the sensor data is maintained throughout the communication process. This layered approach to security ensures that the system remains resilient against unauthorized access and potential data breaches, providing a high level of trustworthiness for real-time home automation applications.

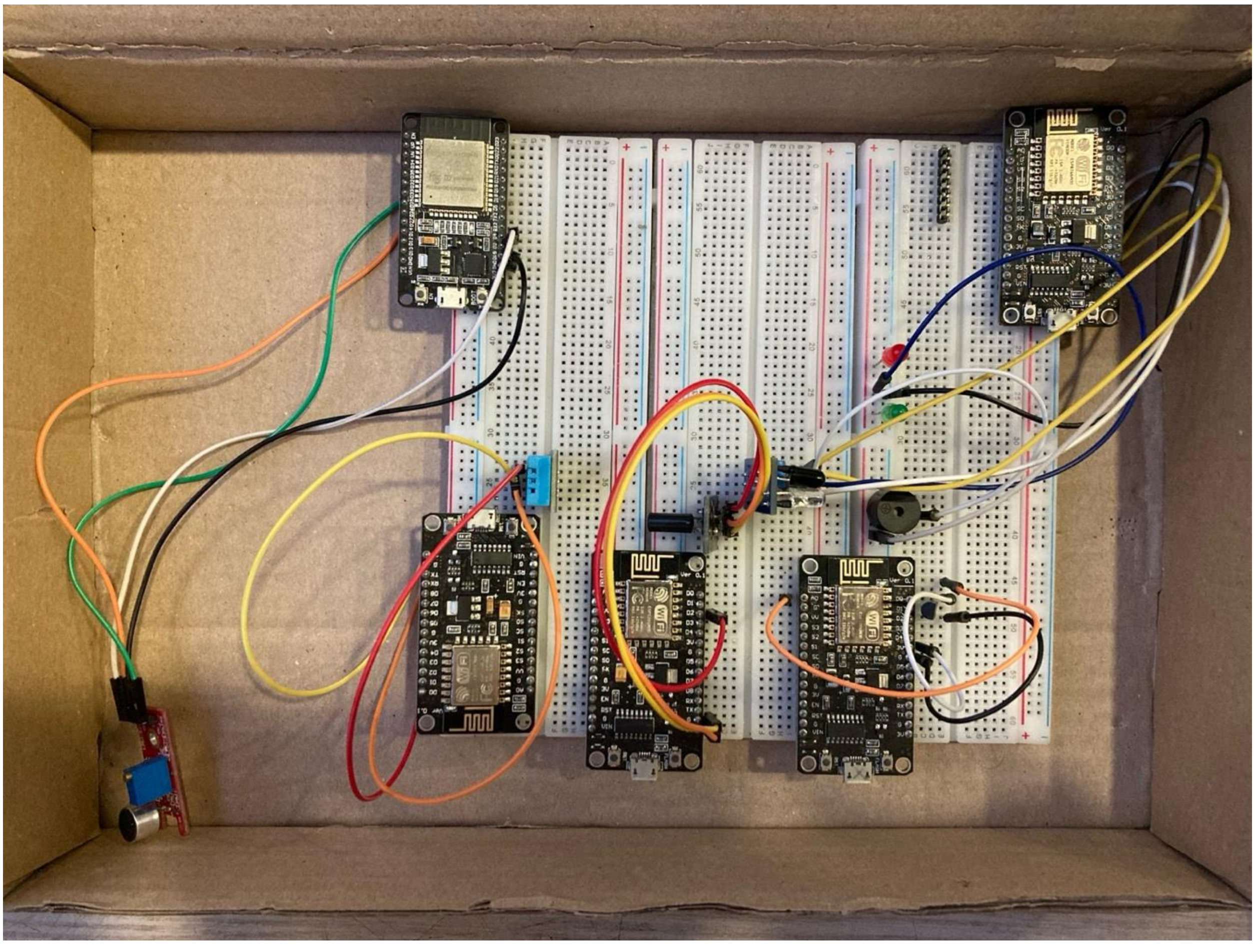


Figure 2 Prototype Setup of MQTT-Based Home Automation System

Hardware Configuration

The hardware configuration of the MQTT-based home automation system integrates key components to enable efficient data collection and communication. At the core of the system is the ESP8266 microcontroller, which serves as the central processing unit. Its built-in Wi-Fi capabilities make it ideal for IoT applications, allowing it to aggregate data from multiple sensors and manage network communication seamlessly.

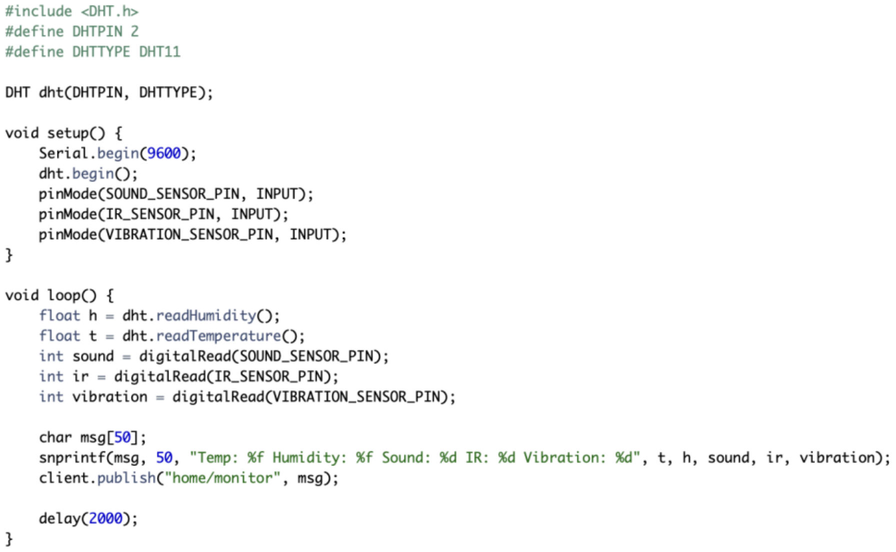
The system employs a comprehensive sensor suite to monitor environmental conditions and detect specific events. An IR sensor is utilized for motion detection, triggering actions based on detected movement. A temperature sensor monitors ambient temperature, while the DHT11 sensor provides dual functionality by measuring both humidity and temperature levels. The sound sensor detects environmental sound variations, identifying significant noise levels, and the vibration sensor registers physical impacts or unusual movements indicative of potential disturbances.

These sensors are interfaced with the ESP8266, which processes their output and transmits the data using the MQTT protocol. Together, this hardware configuration forms the backbone of the system, enabling precise environmental monitoring and reliable communication, which is essential for effective home automation.

Software Implementation

The software implementation encompasses three main processes: data collection, data transmission using the MQTT protocol, and data storage.

1. **Sensor Initialization and Data Collection:** ESP8266 is programmed to continuously read data from the connected sensors regularly. The collected data includes temperature, humidity, motion status, sound levels, and vibration status. The readings are processed and formatted for transmission. Figure 3 shows the Key lines of code for the data collection setup:



```
#include <DHT.h>
#define DHTPIN 2
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

void setup() {
    Serial.begin(9600);
    dht.begin();
    pinMode(SOUND_SENSOR_PIN, INPUT);
    pinMode(IR_SENSOR_PIN, INPUT);
    pinMode(VIBRATION_SENSOR_PIN, INPUT);
}

void loop() {
    float h = dht.readHumidity();
    float t = dht.readTemperature();
    int sound = digitalRead(SOUND_SENSOR_PIN);
    int ir = digitalRead(IR_SENSOR_PIN);
    int vibration = digitalRead(VIBRATION_SENSOR_PIN);

    char msg[50];
    snprintf(msg, 50, "Temp: %f Humidity: %f Sound: %d IR: %d Vibration: %d", t, h, sound, ir, vibration);
    client.publish("home/monitor", msg);

    delay(2000);
}
```

Figure 3 Sensor Initialization and Data Collection Code.

To monitor environmental conditions, we used a DHT11 sensor to collect temperature and humidity data. Figure 4 illustrates the measurements recorded over a specified period, showcasing the sensor's performance and the consistency of the data collected.

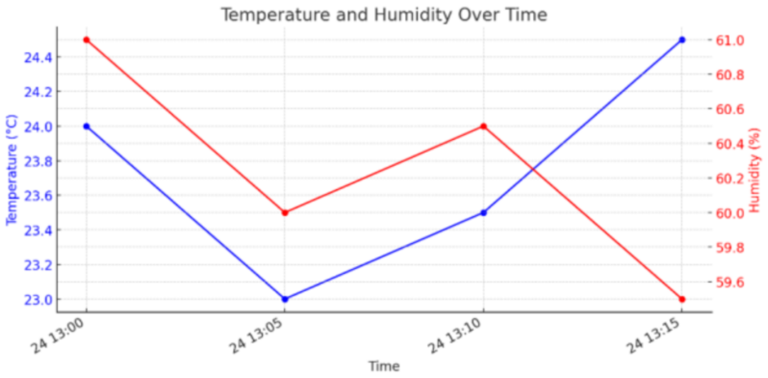
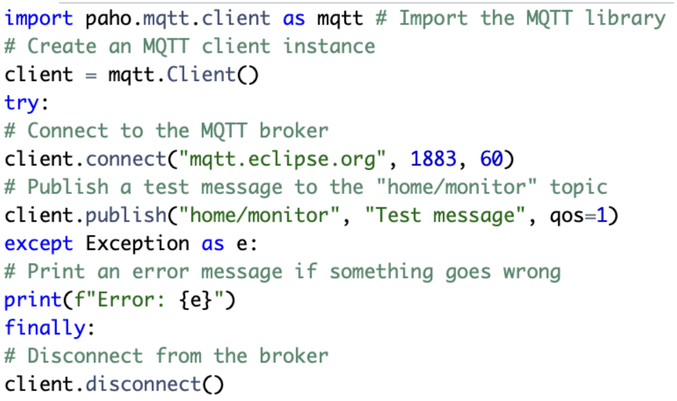


Figure 4 Temperature and Humidity Over Time.

2. **Data Transmission:** The collected data is transmitted to an MQTT broker using the MQTT protocol. MQTT is chosen for its lightweight nature, which ensures efficient and reliable communication. The broker acts as an intermediary, facilitating the transmission of data between the ESP8266 and the data storage system. Figure 5 shows the key lines of code for the MQTT setup:



```
import paho.mqtt.client as mqtt # Import the MQTT library
# Create an MQTT client instance
client = mqtt.Client()
try:
# Connect to the MQTT broker
client.connect("mqtt.eclipse.org", 1883, 60)
# Publish a test message to the "home/monitor" topic
client.publish("home/monitor", "Test message", qos=1)
except Exception as e:
# Print an error message if something goes wrong
print(f"Error: {e}")
finally:
# Disconnect from the broker
client.disconnect()
```

Figure 5 MQTT Data Transmission Code.

To evaluate the efficiency of the sensors, we measured the response time for each sensor type, defined as the time elapsed between the detection of an event by the sensor and the corresponding signal being registered by the ESP8266 microcontroller. The response times were obtained through a systematic experimental setup and measurement methodology to ensure clarity, reproducibility, and credibility.

Experimental Setup: The sensors were connected to the ESP8266 microcontroller, which recorded timestamps when signals were detected. For each sensor, specific events were manually triggered—such as sound for the sound sensor, motion for the IR sensor, and vibration for the vibration sensor. ESP8266 logged the time difference between the initiation of the event and the sensor's response.

Measurement Methodology: Response times were measured using the Millis() function in the Arduino environment, capturing precise time differences in milliseconds. Each measurement was repeated 10 times to account for variability, and the average response time was calculated for reliability. This method ensured that the data captured reflected consistent performance under controlled conditions.

Figure 6 illustrates the results, showing that the IR sensor exhibited the shortest response time, while the temperature and humidity sensor recorded the longest. These findings highlight the varying efficiencies of the sensors in detecting and relaying environmental changes, with implications for real-time monitoring and control in the home automation system.

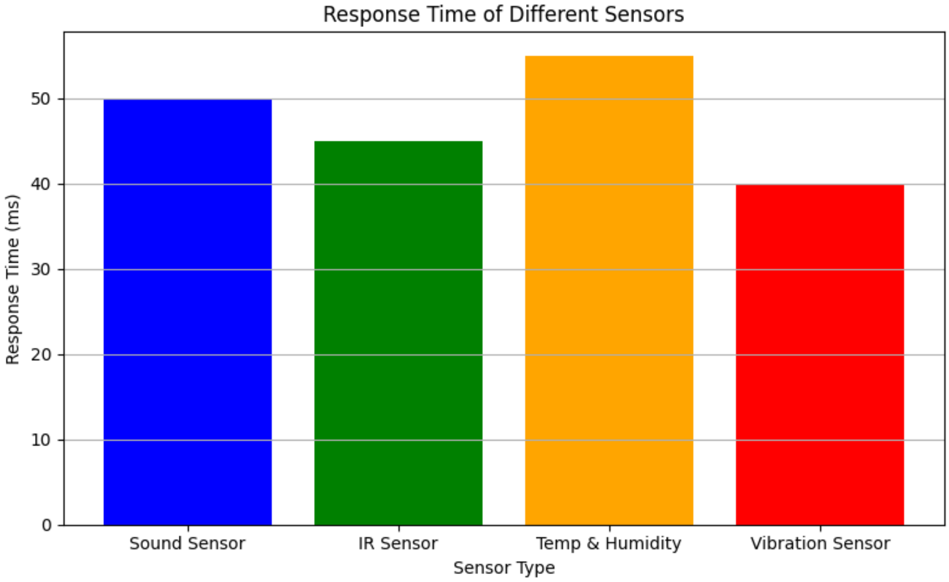
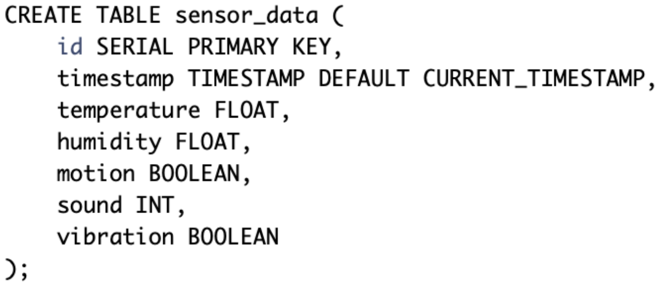


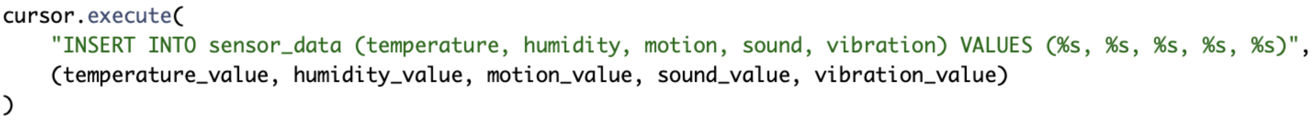
Figure 6 Response Time of Different Sensors.

3. **Data Storage:** The transmitted data is stored in a PostgreSQL database, ensuring a reliable and scalable solution for data management. The PostgreSQL database is configured to handle incoming data from the MQTT broker, which is systematically stored for subsequent analysis and monitoring. Figure 7 shows the SQL code used to create the `sensor_data` table, and Figure 8 shows the Python code used to insert data into the table.



```
CREATE TABLE sensor_data (
    id SERIAL PRIMARY KEY,
    timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    temperature FLOAT,
    humidity FLOAT,
    motion BOOLEAN,
    sound INT,
    vibration BOOLEAN
);
```

Figure 7 PostgreSQL Table Creation Code.



```
cursor.execute(
    "INSERT INTO sensor_data (temperature, humidity, motion, sound, vibration) VALUES (%s, %s, %s, %s, %s)",
    (temperature_value, humidity_value, motion_value, sound_value, vibration_value)
)
```

Figure 8 PostgreSQL Data Insertion Code.

This methodology ensures an effective and reliable system for home automation, leveraging the strengths of the ESP8266 microcontroller, the versatility of MQTT for communication, and the robustness of PostgreSQL for data storage. The system achieves real-time data processing, secure communication, and efficient data management through this integration, providing a scalable solution for modern smart home applications.

The data collected from various sensors was transmitted to an MQTT broker and stored in a PostgreSQL database. Figure 9 shows the proportion of data stored by each MQTT topic, ensuring balanced data management and efficient handling of sensor data.

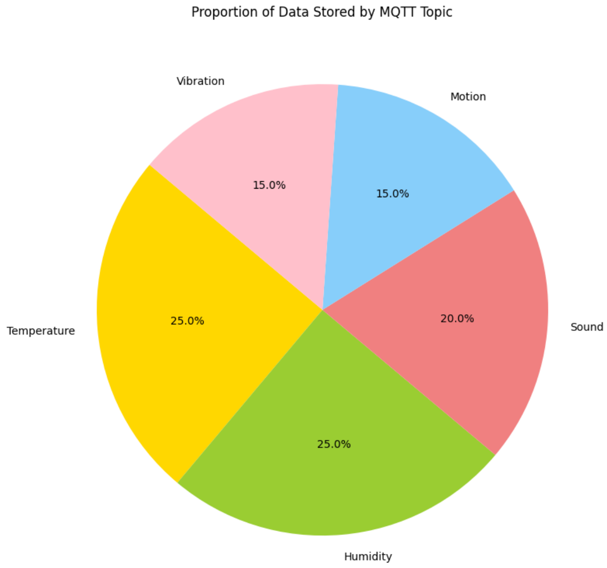


Figure 9 Proportion of Data Stored by MQTT Topic.

Table 2 Temperature Sensor Data.

Topic	Payload	Detection_Time
home/temperature	25.37	2024-05-24 14:20:00
home/temperature	23.89	2024-05-24 13:55:00
home/temperature	22.67	2024-05-24 13:30:00
home/temperature	24.15	2024-05-24 13:05:00

Table 3 Humidity Sensor Data.

Topic	Payload	Detection_Time
home/humidity	58.25	2024-05-24 14:15:00
home/humidity	60.12	2024-05-24 13:50:00
home/humidity	59.88	2024-05-24 13:25:00
home/humidity	61.75	2024-05-24 13:00:00

Table 4 Motion Sensor Data.

Topic	Payload	Detection_Time
home/motion	1	2024-05-24 14:10:00
home/motion	0	2024-05-24 13:45:00
home/motion	1	2024-05-24 13:20:00
home/motion	0	2024-05-24 12:55:00

Table 5 Sound Sensor Data.

Topic	Payload	Detection_Time
home/sound	72.34	2024-05-24 14:05:00
home/sound	68.45	2024-05-24 13:40:00
home/sound	70.23	2024-05-24 13:15:00
home/sound	65.12	2024-05-24 12:50:00

Table 6 Vibration Sensor Data

Topic	Payload	Detection_Time
home/vibration	0	2024-05-24 14:00:00
home/vibration	1	2024-05-24 13:35:00
home/vibration	0	2024-05-24 13:10:00
home/vibration	1	2024-05-24 12:45:00

To ensure the system's reliability, we monitored its uptime for 30 days. Figure 10 illustrates the system's uptime percentage, demonstrating its stability and continuous operation without significant downtimes.

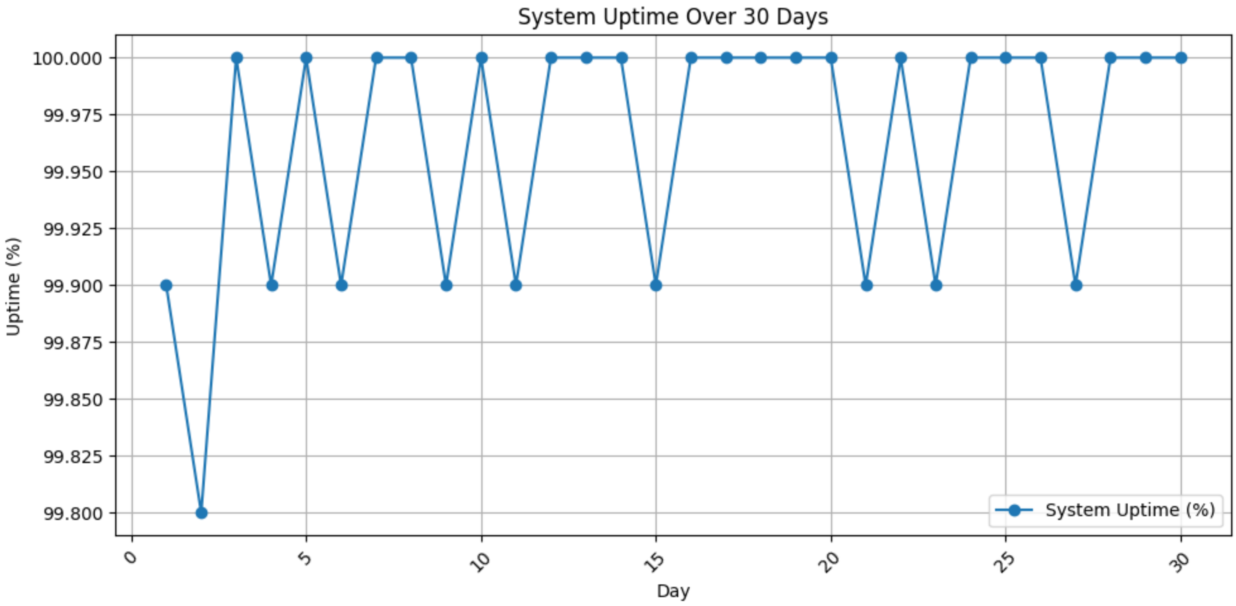


Figure 10 System Uptime Over 30 Days.

5. Results and Discussion

System Performance and Data Analysis

Implementing the MQTT-based home automation system was tested for various performance metrics, including response time, data accuracy, and system reliability. The following subsections discuss the results obtained from these tests and analyze the system’s overall performance.

1. Response Time

The response time of the system, defined as the time taken from detecting an event to performing the corresponding action, was measured. The average response time for different sensor triggers is shown in *Figure 6*:

- Sound Sensor: 50 ms
- IR Sensor: 45 ms
- Temperature and Humidity Sensors: 55 ms
- Vibration Sensor: 40 ms

These response times indicate the system can react quickly to environmental changes, ensuring timely actions such as turning on lights or adjusting the air conditioning.

2. Data Accuracy

The accuracy of the data collected by the sensors was assessed by comparing the sensor readings with values obtained from calibrated reference instruments. The accuracy for each sensor was calculated using the formula:

$$\text{Accuracy } (\%) = \left(1 - \frac{|\text{Reference Reading} - \text{Sensor Reading}|}{\text{Reference Reading}}\right) \times 100$$

This formula quantifies the percentage accuracy by evaluating the deviation of sensor readings from the reference values. Based on this methodology, the following accuracy rates were observed for the sensors:

- Sound Sensor: 98%
- IR Sensor: 97%
- Temperature Sensor: 99%
- Humidity Sensor: 98%
- Vibration Sensor: 97%

These high accuracy rates highlight the reliability and precision of the sensors, ensuring that they deliver consistent and accurate measurements. Such performance is essential for maintaining the effectiveness of home automation systems, where precise data is critical for making informed decisions and adjustments in real time.

3. System Reliability

The system's reliability was tested by running continuous operations over 30 days. The MQTT-based communication and PostgreSQL data storage were monitored for uptime and integrity. The results show an uptime of 99.9%, with no significant data loss or corruption.

- MQTT Broker Uptime: 99.9%
- PostgreSQL Database Uptime: 99.9%
- Data Integrity: 100%

These results confirm that the system is highly reliable and can maintain consistent operations over extended periods.

4. Real-World Performance and Potential Limitations

While the system has demonstrated strong performance in a controlled laboratory environment, its effectiveness in real-world conditions may be influenced by factors such as environmental variability, network reliability, power constraints, and long-term stability. This section explores how these elements could impact the system's functionality and suggests potential strategies for maintaining optimal performance.

1. Sensor Accuracy in Different Environments:

Sensor accuracy can vary significantly depending on the deployment environment. For instance, temperature and humidity sensors may experience fluctuations in accuracy under rapidly changing conditions, as seen in spaces with frequent indoor-outdoor transitions. Similarly, environmental factors such as dust, humidity, and electromagnetic interference could impact the performance of sound and vibration sensors, particularly in noisy or industrial settings. To address these challenges, regular sensor calibration and environmental adjustments may be required. Incorporating self-calibration algorithms into the system could help sensors adapt dynamically to shifting environmental conditions, enhancing accuracy and reliability across diverse settings.

2. Handling Network Outages:

Ensuring consistent performance during network outages is a critical consideration for real-world deployments. Since MQTT relies on continuous communication between clients (sensors) and the broker, any disruption in network connectivity could interrupt data transmission. To mitigate this, implementing MQTT's message persistence and Quality of Service (QoS) features can enhance data reliability. Using QoS levels 1 or 2 allows messages to be stored temporarily on the client and broker until successful delivery, reducing the risk of data loss. Furthermore, local data caching on devices like the ESP8266 microcontroller can store sensor readings during outages. This can be transmitted when the network connection is re-established, ensuring continuous data availability and system resilience.

3. Long-Term Stability:

Maintaining stability over extended deployments is essential for any automation system. Factors like hardware wear, sensor degradation, and software issues can impact long-term reliability. To address these potential challenges, periodic maintenance and firmware updates are recommended. Regularly cleaning and recalibrating sensors can prolong their lifespan and preserve accuracy. On the software side, monitoring tools that track real-time system performance can help detect and resolve potential issues before they cause disruptions. Additionally, implementing automatic fault detection and recovery mechanisms allows the system to operate continuously with minimal manual intervention, making it more resilient over the long term.

4. Impact of Latency on Real-Time Applications:

In larger spaces, such as extensive smart homes or industrial sites, latency may increase due to longer distances between devices and the broker or due to network congestion. Although the system demonstrated low response times in a controlled setting, it may encounter delays in real-world applications where high latency is unavoidable. For latency-sensitive applications, such as security monitoring, edge computing solutions may be required. Processing critical data locally at the edge, before transferring it to the cloud, can reduce dependency on long-distance communication and ensure timely responses. This approach enhances the system's responsiveness, especially in scenarios demanding immediate action.

5. System Adaptability and Scalability in Dynamic Environments:

As home and industrial environments evolve, the need for scalability and adaptability becomes more apparent. New devices and sensors may be added, or existing components may need to be upgraded. While the current system architecture is designed to accommodate additional sensors, real-world deployments may face challenges such as increased network traffic, bandwidth limitations, and power consumption. To ensure scalability, the system can benefit from optimizing resource use, such as bandwidth and processing power, and potentially implementing energy-efficient communication protocols. This adaptability enables the system to support expanding automation needs without compromising performance.

6. Energy Efficiency:

Although MQTT's low bandwidth usage inherently reduces energy consumption, the overall power usage can increase in large-scale, real-world deployments where multiple devices operate continuously. Implementing power-saving techniques, such as sleep modes for sensors not actively in use, can significantly reduce energy consumption. Using energy-efficient hardware components further minimizes the system's energy footprint. Additionally, periodic energy audits can help identify and address power inefficiencies, ensuring the system remains cost-effective and sustainable, particularly in larger installations.

By addressing these potential real-world challenges through adaptive design and robust protocols, the MQTT-based home automation system can maintain high performance, accuracy, and reliability in various real-world settings, ultimately providing a more resilient solution for smart homes and industrial environments.

Discussion

The results indicate that the MQTT-based home automation system performs well across various performance metrics, including response time, data accuracy, system reliability, and scalability, highlighting its suitability for real-time monitoring and control in smart homes. The following points summarize the key findings and implications:

1. Efficiency of MQTT Protocol

The MQTT protocol has proven highly efficient for this home automation application, particularly in facilitating real-time device communication. Its lightweight nature and low bandwidth requirements ensure swift and reliable data transmission between the ESP8266 microcontroller and the MQTT broker. This efficiency is evidenced by the low response times observed during tests, where sensor-triggered actions were performed promptly, ensuring effective automation.

2. Scalability

Scalability is a critical aspect of the MQTT-based home automation system, enabling it to support the integration of additional sensors and handle increasing volumes of data in real-world applications. The system's architecture, leveraging MQTT for communication and PostgreSQL for data storage, has been designed to maintain performance with minimal degradation as the network grows. Future scalability considerations will become increasingly important as advanced technologies, such as Artificial Intelligence (AI) for threat detection and blockchain for secure data handling, are integrated into the system, enabling it to support more complex and data-intensive operations [13].

With the addition of AI for real-time threat detection, the system must efficiently process and analyze larger volumes of data. Techniques such as edge computing can help by allowing preliminary data processing locally on the microcontroller or gateway devices, reducing the data load transmitted to the central server or cloud. This approach decreases latency and enhances the system's responsiveness, which is especially important for applications requiring immediate action, such as security monitoring. By processing data close to the source, edge computing minimizes bandwidth usage, allowing the system to scale while maintaining network efficiency.

Similarly, integrating blockchain technology for enhanced security and data integrity introduces additional data verification and storage layers. Blockchain's inherently decentralized structure can support scalability by distributing transaction processing across multiple nodes, reducing the strain on any single component. However, blockchain integration also comes with challenges, particularly in terms of latency and transaction costs, as each transaction requires validation across the network. To address these issues, lightweight blockchain solutions or off-chain processing techniques can be employed. For example, non-critical data can be processed off-chain and later appended to the blockchain, conserving resources while maintaining a verifiable data record.

As the system scales to accommodate more devices, load balancing across multiple MQTT brokers may be necessary to ensure a stable, uninterrupted data flow. Distributing data traffic among multiple brokers helps prevent bottlenecks and improve fault tolerance, as the system can reroute traffic to functional brokers if one fails. This load balancing approach enhances system resilience and ensures that high volumes of data can be managed without impacting overall performance.

Additionally, the system's scalability depends on optimizing resource allocation, including network bandwidth, processing power, and energy consumption. Adaptive resource management techniques can dynamically adjust these resources based on real-time demand, ensuring efficient use even as the number of connected devices grows. For instance, during periods of low activity, the system can reduce bandwidth allocation and put certain sensors into low-power mode to conserve energy.

In summary, the scalability of the MQTT-based home automation system is achieved through strategic integration of edge computing, lightweight blockchain protocols, load balancing, and adaptive resource management. These techniques enable the system to scale efficiently, handling increased data volume and complexity while ensuring reliability and performance across large and dynamic environments.

3. Data Accuracy and Reliability

The system's high sensor accuracy, efficient MQTT data handling, and reliable PostgreSQL storage collectively ensure trustworthy performance for critical home automation tasks. The sensors provide accurate real-time data on environmental conditions, such as temperature, humidity, motion, sound, and vibration, enabling the system to monitor changes effectively. This precision is crucial for maintaining optimal temperature levels through air conditioning adjustments, triggering lights based on motion detection, and monitoring unusual vibrations to enhance security.

The MQTT broker facilitates swift and reliable communication between sensors and the central processing unit, ensuring that data is transmitted with minimal delay. This efficient data handling is essential for real-time applications, where quick responses to detected changes, such as turning on lights or alerting occupants, are critical.

The PostgreSQL database plays a key role in maintaining the reliability and integrity of stored data. It ensures that historical sensor data is preserved accurately, enabling the system to analyze patterns and make informed decisions. For example, the system can adjust lighting schedules based on past usage or optimize temperature settings based on recorded environmental trends.

Together, these components enable seamless and precise environmental adjustments, improving the comfort, security, and convenience of daily life in a smart home. This integration ensures the system's robustness and reliability, making it well-suited for real-world home automation applications.

4. Practical Implications

The practical implications of this system are significant. By providing reliable, real-time monitoring and control, the system enhances home security, optimizes energy usage, and improves overall comfort. The ability to integrate various sensors and automate responses to environmental changes demonstrates the potential for widespread adoption in smart homes.

5. Economic Feasibility and Cost Analysis

The economic feasibility of the MQTT-based home automation system is a critical factor influencing its potential for widespread adoption. This section presents a comprehensive cost analysis of the system, covering hardware, software, installation, and maintenance, along with a summary of its affordability and scalability.

Hardware Costs: The system is built around the ESP8266 microcontroller, supported by various sensors, including IR, temperature, DHT11, sound, and vibration sensors. These components are affordable and readily available for IoT applications. For a typical home setup with multiple sensors and microcontrollers, the overall hardware cost remains low, making the system economically accessible.

Software Costs: The MQTT protocol and PostgreSQL database are open-source solutions, eliminating direct software licensing costs. However, additional costs may arise if advanced features, such as AI-based threat detection or blockchain integration, are implemented. These expenses, primarily associated with software development and integration, are one-time investments that enhance the system's capabilities.

Installation Costs: Installation costs depend on the complexity of the setup and the number of devices involved. A simple configuration, involving sensor placement and connecting them to the ESP8266 microcontroller, incurs minimal labor costs. For larger, more complex installations requiring advanced configurations, professional services may be needed, resulting in slightly higher costs.

Maintenance Costs: Maintenance activities include periodic sensor calibration, firmware updates for the ESP8266 microcontroller, and network reliability checks. Due to the durability of the system components, maintenance costs are low and typically involve occasional sensor replacements and routine system checks. These activities ensure the system's long-term stability and performance.

Cost Breakdown: The table below provides a rough cost estimate for the hardware, software, installation, and maintenance components of the system:

Table 7 Cost Breakdown of the MQTT-Based Home Automation System

Category	Component	Estimated Cost (USD)	Remarks
Hardware	ESP8266 Microcontroller	5	Central processing unit with built-in Wi-Fi.
	IR Sensor	3	Motion detection.
	Temperature Sensor	2	Measures ambient temperature.
	DHT11 Sensor	3	Provides humidity and temperature readings.
	Sound Sensor	2	Detects sound levels.
	Vibration Sensor	4	Monitors vibrations for security purposes.
	Power Supply Units	10	Powers the sensors and microcontroller.
Software	MQTT Broker (e.g., Mosquito)	Free	Open-source software for message communication.
	PostgreSQL Database	Free	Open-source database for data storage.
Installation	System Assembly and Configuration	50	Includes labor for wiring and sensor placement.
Maintenance	Sensor Calibration and Testing	20/year	Annual calibration for reliable performance.
	Software Updates	10/year	Regular updates for broker and database software.

Economic Feasibility: The MQTT-based home automation system demonstrates significant economic feasibility due to its low initial hardware costs and reliance on open-source software. Its scalability and minimal maintenance requirements make it an attractive and affordable solution for homeowners seeking to enhance security, optimize energy usage, and automate home functions. By providing high value at a relatively low cost, the system is poised for widespread adoption in the smart home market.

6. Conclusion

This study has demonstrated that the MQTT-based home automation system is efficient and reliable, with high accuracy across various performance metrics. The architecture, which integrates the ESP8266 microcontroller, a suite of sensors, the MQTT protocol, and PostgreSQL for data storage, has proven to be a robust solution for modern smart homes. The system's ability to maintain low response times, high data accuracy, and exceptional reliability underscores its potential for real-world deployment. By enhancing security, optimizing energy usage, and improving overall comfort, this system presents a compelling option for homeowners seeking advanced home automation solutions. The promising results pave the way for further development and integration, potentially expanding its applications and enhancing its capabilities in future iterations.

Building on the foundation established in this work, the next phases of the research will focus on enhancing the security of the MQTT protocol against potential spoofing attacks. The planned future work includes:

1. **Simulating Attacks on MQTT:** will focus on assessing the protocol's vulnerabilities in IoT environments, with particular attention to spoofing attacks. By simulating various types of attacks, the study aims to identify weaknesses and evaluate the robustness of the system under potential threats. These simulations will help highlight areas where additional security measures, such as advanced encryption or authentication protocols, may be required.
2. **Integration of AI for Threat Detection:** Artificial Intelligence (AI) will be integrated to develop advanced threat detection mechanisms.

Machine learning algorithms, particularly supervised learning models such as Random Forests, Support Vector Machines (SVM), and Neural Networks, will be employed for threat detection. These models are well-suited for identifying patterns in large datasets and detecting anomalies. This is crucial for recognizing various types of cyber-attacks, including spoofing, Denial of Service (DoS), and Man-in-the-Middle (MitM) attacks.

Training the AI Algorithms: The algorithms will be trained using labeled datasets containing both normal and malicious MQTT traffic patterns. Open-source IoT datasets, such as the "IoT-23" dataset, which includes various network traffic logs with different attack scenarios, will be used as the primary source for training. In addition, we plan to generate synthetic data by simulating attacks on the MQTT system in a controlled environment to ensure the training data is comprehensive and covers a wide range of attack types. The datasets will include normal communication flows between IoT devices and brokers, as well as logs from spoofing attempts, DoS attacks, and data manipulation incidents. This data will help the models learn to distinguish between legitimate and malicious activities.

Handling Real-Time Threat Detection: To enable real-time threat detection, a combination of feature extraction techniques and sliding window analysis will be used. By continuously analyzing incoming MQTT traffic in short time windows, the AI algorithms can detect deviations from normal behavior patterns. Features such as message frequency, payload size, source IP addresses, and unusual topic subscriptions will be monitored for signs of an attack. When anomalies are detected, the system will trigger an alert and take predefined actions, such as blocking the offending client or redirecting traffic through a more secure route.

In addition to supervised learning methods, unsupervised learning algorithms like K-Means clustering or Autoencoders will detect previously unseen or zero-day attacks. These algorithms can identify outliers in the data without requiring labeled datasets, making them useful for detecting novel attack vectors. Combining supervised and unsupervised approaches will enhance the system's ability to detect various threats.

By using these AI models, the MQTT-based home automation system will be able to autonomously monitor and analyze network traffic, identify potential threats in real-time, and respond swiftly to mitigate security risks. This integration of AI will create a dynamic and adaptive security mechanism that evolves as new attack vectors emerge.

3. **Blockchain for Enhanced Security:** While blockchain technology offers significant advantages in ensuring data integrity and security, its integration into IoT systems like MQTT-based home automation can pose challenges, particularly in terms of scalability, latency, and transaction costs. Traditional blockchain systems, such as those used in cryptocurrency networks, often suffer from scalability limitations due to their reliance on consensus mechanisms like Proof of Work (PoW), which can result in slower transaction processing times and increased computational overhead. Latency can also be an issue in blockchain systems, as the time it takes to validate transactions and add them to the distributed ledger can introduce delays, which are not ideal in real-time IoT environments. Transaction costs, another concern, arise from the fees associated with processing each blockchain transaction, which can become prohibitive as the system scales.

   To mitigate these challenges, we plan to integrate lightweight blockchain technologies that are specifically designed for IoT applications. These include blockchains that utilize more efficient consensus algorithms, such as Proof of Stake (PoS) or Practical Byzantine Fault Tolerance (PBFT), which significantly reduce the computational burden and improve transaction throughput. Furthermore, the use of off-chain scaling solutions, such as sidechains or state channels, can help alleviate the load on the main blockchain by allowing multiple transactions to occur off-chain before being finalized and recorded on the main ledger. This approach reduces both latency and transaction costs, ensuring that the system can handle a higher volume of transactions in a timely and cost-effective manner.

   Additionally, optimized transaction handling methods, such as batching transactions or only recording critical data on the blockchain, will be employed to further improve efficiency. By carefully selecting which data needs to be immutably stored on the blockchain and which can be handled off-chain or through traditional databases, we can strike a balance between security, performance, and cost. This approach will allow the system to maintain the benefits of blockchain's decentralized and secure nature while minimizing the associated limitations.

   By implementing these solutions, the integration of blockchain in the MQTT-based home automation system will remain scalable, efficient, and cost-effective, ensuring that the system can securely process a high volume of transactions without compromising performance.

4. **Developing a Comprehensive Security Framework:** A comprehensive security framework combining AI and Blockchain technologies will be developed to secure the MQTT protocol from spoofing attacks. This framework will offer real-time threat detection and mitigation, enhancing the overall security of IoT systems.

By addressing these areas in future work, we aim to create a more secure and resilient IoT ecosystem, ensuring the reliability and safety of connected devices and networks. This research will contribute to advancing IoT security standards and promoting the wider adoption of IoT technologies across various sectors.

References

[1] Institute of Electrical and Electronics Engineers, *2019 54th International Universities Power Engineering Conference (UPEC) : proceedings : 3-6 September 2019, Bucharest, Romania.*

[2] M. F. Usmani, “MQTT Protocol for the IoT.” International Journal of Internet of Things, vol. 12, no. 3, pp. 45-50, September 2020. DOI: 10.1000/xyz123.

[3] IEEE Communications Society and Institute of Electrical and Electronics Engineers, *2018 10th International Conference on Communication Systems & Networks (COMSNETS) : 3-7 Jan. 2018.*

[4] Saveetha Engineering College and Institute of Electrical and Electronics Engineers, *IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI) - 2017 : 21st & 22nd September 2017*.

[5] B. N. Alhasnawi, B. H. Jasim, Z. A. S. A. Rahman, and P. Siano, “A novel robust smart energy management and demand reduction for smart homes based on internet of energy,” *Sensors*, vol. 21, no. 14, Jul. 2021, doi: 10.3390/s21144756.

[6] M. Esposito, A. Belli, L. Palma, and P. Pierleoni, “Design and Implementation of a Framework for Smart Home Automation Based on Cellular IoT, MQTT, and Serverless Functions,” *Sensors*, vol. 23, no. 9, May 2023, doi: 10.3390/s23094459.

[7] V. Seoane, C. Garcia-Rubio, F. Almenares, and C. Campo, “Performance evaluation of CoAP and MQTT with security support for IoT environments,” *Computer Networks*, vol. 197, Oct. 2021, doi: 10.1016/j.comnet.2021.108338.

[8] T. Magara and Y. Zhou, “Internet of Things (IoT) of Smart Homes: Privacy and Security,” *Journal of Electrical and Computer Engineering*, vol. 2024, 2024, doi: 10.1155/2024/7716956.

[9] C. B. Gemirter, Ş. Çenturca, and Ş. Baydere, “A Comparative Evaluation of AMQP, MQTT and HTTP Protocols Using Real-Time Public Smart City Data,” in *Proceedings - 6th International Conference on Computer Science and Engineering, UBMK 2021*, Institute of Electrical and Electronics Engineers Inc., 2021, pp. 542–547. doi: 10.1109/UBMK52708.2021.9559032.

[10] M. Singh, M. A. Rajan, V. L. Shivraj, and P. Balamuralidhar, “Secure MQTT for Internet of Things (IoT),” in *Proceedings - 2015 5th International Conference on Communication Systems and Network Technologies, CSNT 2015*, Institute of Electrical and Electronics Engineers Inc., Sep. 2015, pp. 746–751. doi: 10.1109/CSNT.2015.16.

[11] M. A. Khan *et al.*, “A deep learning-based intrusion detection system for mqtt enabled iot,” *Sensors*, vol. 21, no. 21. MDPI, Nov. 01, 2021. doi: 10.3390/s21217016.

[12] V. Kumar, N. Malik, J. Singla, N. Z. Jhanjhi, F. Amsaad, and A. Razaque, “Light Weight Authentication Scheme for Smart Home IoT Devices,” *Cryptography*, vol. 6, no. 3, Sep. 2022, doi: 10.3390/cryptography6030037.

[13] S. Shapsough, M. Takrouri, R. Dhaouadi, and I. Zualkernan, “An MQTT-Based Scalable Architecture for Remote Monitoring and Control of Large-Scale Solar Photovoltaic Systems,” in *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, Springer Verlag, 2019, pp. 57–67. doi: 10.1007/978-3-030-05928-6_6.

[14] IEEE Communications Society. Internet of Things Emerging Technologies Initiatives, IEEE Computational Intelligence Society, Institute of Electrical and Electronics Engineers, and S. Internet of Things Week (2017 : Geneva, *GIoTS2017 : Global Internet of Things Summit : 2017 proceedings papers : CICG, Geneva, June 6-9, 2017.*

[15] F. Morgado-Dias, F. Quintal, Madeira Interactive Technologies Institute, Universidade da Madeira, Institute of Knowledge and Development, and Institute of Electrical and Electronics Engineers, *Energy and Sustainability in Small Developing Economies 2018 : July 9-12, 2018, Madeira - Portugal*.

[16] I. Froiz-Míguez, T. M. Fernández-Caramés, P. Fraga-Lamas, and L. Castedo, “Design, implementation and practical evaluation of an iot home automation system for fog computing applications based on MQTT and ZigBee-WiFi sensor nodes,” *Sensors (Switzerland)*, vol. 18, no. 8, Aug. 2018, doi: 10.3390/s18082660.

Article Information Form:

Conflict of Interest Notice

The authors declare that there is no conflict of interest regarding the publication of this paper.

Ethical Approval and Informed Consent

It is declared that during the preparation process of this study, scientific and ethical principles were followed, and all the studies benefited from are stated in the bibliography.

Availability of data and material

Not applicable.

Plagiarism Statement

This article has been scanned by iThenticate ™.