Review

Balancing Security and Performance in MQTT-Based IoT Systems: A Review of Adaptive Flow Control, Backpressure, and Wildcard-Intensive Access Control Mechanisms

Nael M Radwan * and Frederick T Sheldon

Posted Date: 3 April 2026

doi: 10.20944/preprints202604.0222.v1

Keywords: MQTT; internet of things (IoT); backpressure; adaptive flow control; congestion control; wildcard subscriptions; authentication; authorization; access control; Quality of Service (QoS); MQTT v5; receive maximum; broker performance; IoT security; publish–subscribe systems; traffic management; scalability; latency; resource optimization; cybersecurity in IoT

 doi:10.20944/preprints202604.0222.v1

Review

Balancing Security and Performance in MQTT-Based IoT Systems: A Review of Adaptive Flow Control, Backpressure, and Wildcard-Intensive Access Control Mechanisms

Nael M Radwan [1,*] and Frederick T Sheldon [2]

[1] Computer Science Department, University of Idaho, Moscow, ID, USA

* Correspondence: nradwan@uidaho.edu

Abstract

The rapid proliferation of the Internet of Things (IoT) has positioned the Message Queuing Telemetry Transport (MQTT) protocol as a fundamental communication standard for large-scale, resource-constrained systems. Despite its lightweight design and scalability advantages, modern MQTT deployments operate under increasingly complex conditions characterized by intensive security enforcement, dynamic traffic patterns, and widespread use of wildcard subscriptions. These factors introduce tightly coupled challenges related to system performance, congestion, and security, which are often addressed independently in existing literature. This review provides a comprehensive and critical analysis of MQTT-based IoT systems, focusing on the interaction between adaptive flow control, backpressure phenomena, security mechanisms, and wildcard-intensive access control strategies. The study synthesizes recent research on authentication, authorization, and encryption techniques, highlighting their impact on computational overhead, latency, and broker load. In parallel, it examines backpressure formation as a system-level phenomenon arising from the imbalance between message arrivals and processing rates, and evaluates existing flow-control mechanisms, including TCP-based approaches, broker-level controls, and MQTT v5 features such as Receive Maximum. Furthermore, the review investigates the role of wildcard subscriptions in scalable topic management, demonstrating their dual effect as both enablers of efficient data aggregation and amplifiers of routing complexity, traffic load, and security risks. The analysis reveals that wildcard usage significantly increases message fan-out and authorization overhead, thereby accelerating congestion and expanding the attack surface in poorly configured systems. A key contribution of this work is the identification of a fundamental gap in the literature: the absence of integrated, cross-layer frameworks that jointly consider security, flow control, and wildcard behavior under realistic IoT workloads. Current approaches remain fragmented, leading to inefficiencies, reduced reliability, and potential vulnerabilities in large-scale deployments. Based on this synthesis, the paper outlines a forward-looking research roadmap that emphasizes security-aware adaptive flow control, wildcard-aware traffic optimization, cross-layer system design, and intelligent (AI-driven) management strategies. These directions are essential for enabling next-generation MQTT systems that are secure, scalable, and resilient in dynamic and adversarial environments.

Keywords: MQTT; internet of things (IoT); backpressure; adaptive flow control; congestion control; wildcard subscriptions; authentication; authorization; access control; Quality of Service (QoS); MQTT v5; receive maximum; broker performance; IoT security; publish–subscribe systems; traffic management; scalability; latency; resource optimization; cybersecurity in IoT

 doi:10.20944/preprints202604.0222.v1

1. Introduction

1.1. Background and Motivation

The rapid evolution of the Internet of Things (IoT) has transformed modern digital infrastructures into highly interconnected ecosystems composed of billions of heterogeneous devices, ranging from smart home sensors to industrial control systems. It is estimated that IoT deployments will exceed tens of billions of connected devices globally, generating massive volumes of real-time data and requiring efficient, scalable, and lightweight communication protocols [1,2]. Among these protocols, the Message Queuing Telemetry Transport (MQTT) protocol has emerged as a dominant standard due to its simplicity, low bandwidth consumption, and publish/subscribe communication model.

MQTT enables asynchronous communication between producers (publishers) and consumers (subscribers) via a centralized broker, decoupling data producers from consumers and allowing flexible topic-based routing. This design makes MQTT particularly suitable for resource-constrained environments such as wireless sensor networks and edge-based IoT systems [3]. However, the same features that contribute to its efficiency—lightweight headers, flexible topic hierarchies, and minimal control overhead—also introduce significant challenges when systems scale or operate under dynamic workloads.

As highlighted in prior studies and supported by the attached work, MQTT-based systems become increasingly vulnerable when deployed in real-world environments characterized by high traffic rates, heterogeneous device capabilities, and complex subscription patterns. In such scenarios, maintaining both **security and performance simultaneously** becomes a fundamental challenge.

1.2. Security Challenges in MQTT-Based IoT Systems

Security remains one of the most critical concerns in IoT deployments [4]. MQTT, by design, does not enforce strong security mechanisms at the protocol level, relying instead on external solutions such as Transport Layer Security (TLS), username/password authentication, and Access Control Lists (ACLs) [5,6]. While these mechanisms enhance confidentiality, integrity, and authentication, they introduce additional computational overhead and communication latency. Recent studies demonstrate that security mechanisms significantly impact system performance, particularly in resource-constrained environments. For instance, TLS handshakes and certificate validation processes increase connection latency and CPU utilization, especially during large-scale device reconnections or burst communication scenarios [7]. Furthermore, authorization mechanisms such as fine-grained ACL enforcement require per-message policy evaluation, which adds processing overhead at the broker level [8].

The attached dissertation and proposal emphasize that misconfigured authentication and authorization policies can lead not only to security vulnerabilities but also to performance degradation and system instability. This highlights a critical insight: **security and performance in MQTT systems are inherently coupled rather than independent dimensions**.

1.3. Backpressure and Flow-Control Limitations

One of the most significant performance challenges in MQTT-based systems is the phenomenon of **backpressure**, which occurs when the rate of message production exceeds the system's ability to process and deliver messages. Under such conditions, broker queues grow, latency increases, and message loss or system failure may occur.

Although MQTT relies on TCP for reliable transport, TCP-level congestion control is insufficient to address application-layer overload in publish/subscribe systems [9]. MQTT v5 introduced new flow-control mechanisms, such as the *Receive Maximum* parameter, which limits the number of in-flight QoS 1/2 messages [10]. However, these mechanisms have several limitations:

- They do not fully regulate **QoS 0 traffic**, which lacks acknowledgment-based control.

 doi:10.20944/preprints202604.0222.v1

- They operate at a **protocol level without awareness of application context or security state**.
- They are insufficient under **burst workloads, wildcard fan-out, or malicious traffic patterns**.

My own experimental results confirm that default MQTT behavior leads to rapid queue growth, increased latency, and degraded reliability under high-load conditions. In contrast, adaptive flow-control strategies can significantly improve system stability.

1.4. Wildcard Subscriptions: Scalability vs. Security Risk

MQTT's topic-based architecture supports wildcard subscriptions using single-level ("+") and multi-level ("#") operators, enabling efficient aggregation of data across large topic hierarchies. This feature is essential for applications such as smart buildings, industrial monitoring, and large-scale sensor networks.

However, wildcard subscriptions introduce critical performance and security challenges:

- **Performance Impact:** Wildcards increase subscription matching complexity, resulting in higher CPU utilization and routing overhead at the broker [11].
- **Traffic Amplification:** A single wildcard subscription can trigger message delivery to a large number of topics, increasing system load.
- **Security Risks:** Broad subscriptions (e.g., "#") may lead to unauthorized data access if access control policies are weak or misconfigured.

The previous work demonstrates that wildcard-intensive environments significantly increase latency, CPU load, and system instability, particularly under high message rates. Additionally, recent security analyses highlight that attackers can exploit wildcard subscriptions to perform data exfiltration or amplify denial-of-service (DoS) attacks [12].

1.5. Interdependence of Security, Wildcards, and Backpressure

A key observation emerging from recent research—and strongly supported by my work—is that MQTT challenges cannot be addressed in isolation. Instead, they form a tightly coupled system:

- **Security mechanisms** increase processing overhead → amplify backpressure.
- **Wildcard subscriptions** increase message fan-out → increase load and routing complexity.
- **Backpressure conditions** affect reliability → potentially weaken security monitoring and control.

Despite this interdependence, most existing studies focus on a single dimension (e.g., security, performance, or routing) without considering their combined effects. This results in fragmented solutions that fail to address real-world IoT deployment challenges [13].

1.6. Research Gap and Contribution of This Review

Based on the analysis of existing literature and the insights derived from the attached dissertation and related work, a clear research gap can be identified:

There is a lack of integrated, cross-layer approaches that jointly consider authentication, authorization, wildcard subscription behavior, and adaptive flow control in MQTT-based IoT systems.

Most prior works:

- Address security without considering performance impact. or
- Propose flow-control mechanisms without incorporating security context. or
- Analyze wildcards without evaluating system-wide congestion effects.

This review aims to bridge this gap by providing:

1. A **comprehensive synthesis** of MQTT security, flow control, and wildcard-related research.
2. A **critical analysis** of their interactions and combined impact on system performance.
3. Identification of **limitations in current approaches**.
4. A set of **future research directions** for building secure, scalable, and adaptive MQTT systems.

 doi:10.20944/preprints202604.0222.v1

1.7. Organization of the Review Article

The remainder of this review article is organized as follows:

- **Section 2: Background and Fundamentals.**

Provides an overview of MQTT architecture, publish/subscribe communication, Quality of Service (QoS) levels, and MQTT v5 enhancements relevant to flow control.

- **Section 3: Security Mechanisms in MQTT Systems.**

Reviews authentication, authorization, encryption techniques, and their performance implications in IoT environments.

- **Section 4: Backpressure and Flow-Control Mechanisms.**

Analyzes congestion behavior in MQTT systems and evaluates existing flow-control approaches, including MQTT v5 features and adaptive strategies.

- **Section 5: Wildcard Subscription Behavior and Optimization.**

Examines the impact of wildcard subscriptions on scalability, routing complexity, and security risks, along with mitigation techniques.

- **Section 6: Integrated Analysis and Literature Gap Discussion.**

Synthesizes the interaction between security, flow control, and wildcard mechanisms, identifying unresolved challenges.

- **Section 7: Future Research Directions.**

Proposes novel directions such as security-aware adaptive flow control, intelligent subscription management, and cross-layer optimization frameworks.

- **Section 8: Conclusions.**

Summarizes the key findings and highlights the importance of integrated approaches for next-generation MQTT-based IoT systems.

2. Background and Fundamentals of MQTT Systems

2.1. Overview of MQTT Architecture

The Message Queuing Telemetry Transport (MQTT) protocol is a lightweight messaging protocol designed for constrained devices and low-bandwidth, high-latency, or unreliable networks. It follows a **publish/subscribe communication model**, where clients interact indirectly through a centralized broker [3,5].

2.1.1. Publish/Subscribe Model

In MQTT, communication is decoupled in three dimensions:

- **Space decoupling**: publishers and subscribers do not know each other.
- **Time decoupling**: communication is asynchronous.
- **Synchronization decoupling**: no blocking between sender and receiver.

The broker acts as an intermediary that receives messages from publishers and forwards them to subscribers based on **topic matching rules**.

 doi:10.20944/preprints202604.0222.v1

2.1.2. MQTT System Architecture

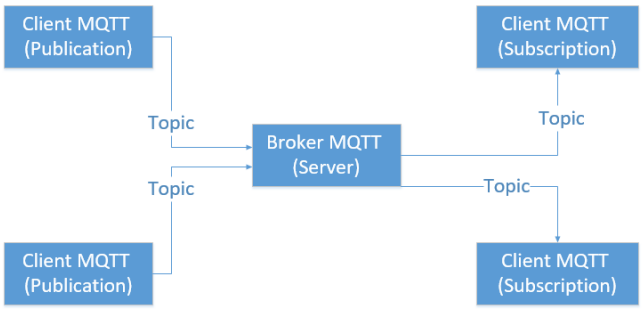

Figure 1. MQTT architecture showing publishers, broker, and subscribers.

The MQTT architecture consists of three main components:

1. **Publisher**: Generates and sends messages to topics.
2. **Broker**: Central server responsible for message routing, filtering, and delivery.
3. **Subscriber**: Receives messages from topics of interest.

The broker performs:

- Topic-based filtering.
- Session management.
- Message queuing and delivery.
- Security enforcement (authentication and authorization).

As highlighted in this research review, the broker becomes a critical bottleneck under high load, especially in wildcard-intensive and authenticated environments [14].

2.2. Topic Hierarchy and Wildcards

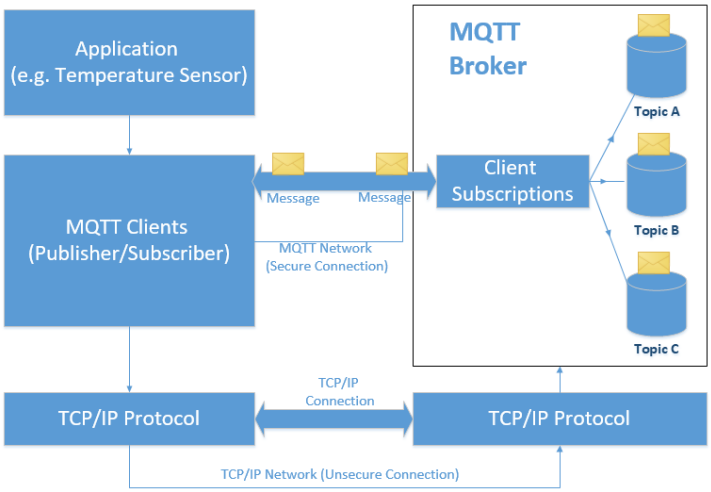

Figure 2. MQTT Architecture: Topic Hierarchy and Wildcards.

MQTT uses a hierarchical topic structure:

home/livingroom/temperature
factory/line1/machine3/status

Wildcard operators:

- + → single-level wildcard.
- # → multi-level wildcard.

Example:

 doi:10.20944/preprints202604.0222.v1

- *home/+/temperature* → matches one level.
- *home/#* → matches entire hierarchy.

While efficient, wildcard usage increases:

- Routing complexity.
- Broker CPU load.
- Security exposure.

2.3. Quality of Service (QoS) Levels

MQTT defines three Quality of Service (QoS) levels that determine message delivery guarantees.

2.3.1. QoS Levels Description

Table 1. MQTT QoS levels comparison.

QoS Level	Name	Guarantee	Mechanism	Overhead
QoS 0	At most once	No guarantee	Fire-and-forget	Low
QoS 1	At least once	Possible duplicates	ACK (PUBACK)	Medium
QoS 2	Exactly once	Guaranteed delivery	4-step handshake	High

2.3.2. QoS Handshake Mechanisms

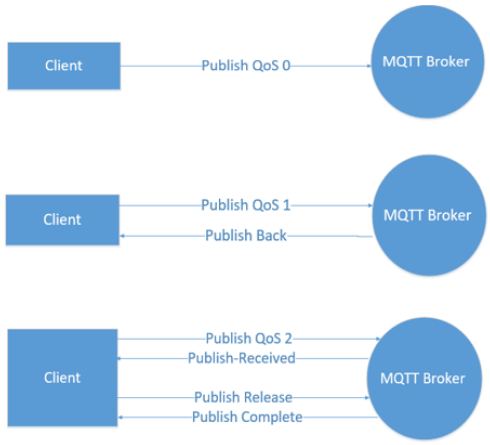

Figure 3. MQTT QoS message exchange mechanisms.

2.3.3. QoS and Performance Trade-Off

QoS levels int roduce a trade-off:

- Higher QoS → better reliability but higher latency and overhead.
- Lower QoS → faster delivery but less reliability.

This trade-off becomes critical under backpressure conditions, where acknowledgment-based QoS (1/2) can amplify congestion due to retransmissions and state tracking [15,16].

2.4. Backpressure in MQTT Systems

Backpressure occurs when: $>service$
Where:

- λin: incoming message rate.
- $\mu service$: processing/service rate.

When this condition holds:

- Queue length increases.

 doi:10.20944/preprints202604.0222.v1

- Latency grows.
- Message loss may occur.

2.4.1. Queue Growth Model

A simplified queue dynamic: $(t+1) = (t) + (t) - (t)$
Where:

- (t): queue size at time t.
- (t): arrival rate.
- (t): service rate.

If $(t) > \mu(t)$, then: $Q(t) \to \infty$
This explains broker overload observed in real deployments and in my experiments.

2.4.2. Impact of Backpressure

Backpressure leads to:

- Increased latency.
- Message drops.
- Broker instability.
- Reduced QoS effectiveness.

Recent studies confirm that MQTT brokers experience **non-linear degradation under overload conditions**, especially in IoT-scale deployments [17–19].

2.5. MQTT v5 Flow-Control Mechanisms

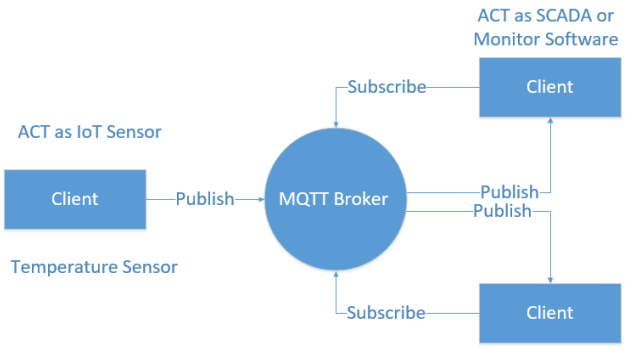

Figure 4. MQTT Work: schematic data flow from the sensor (machine) to devise (machine).

MQTT v5 introduced formal flow-control features to address congestion issues [5,20].

2.5.1. Receive Maximum

The Receive Maximum parameter limits the number of unacknowledged QoS 1/2 messages: $Ninfligh \leq Rmax$
Where:

- $Ninflight$: number of in-flight messages.
- $Rmax$: Receive Maximum.

2.5.2. Send Quota Mechanism

MQTT v5 implements a quota-based model:

$$Quota = Quota - 1 \text{ (on send)}$$
$$Quota = Quota + 1 \text{ (on ACK)}$$

If: $Quota = 0 \to$ sender must stop transmission.

 doi:10.20944/preprints202604.0222.v1

2.5.3. Flow-Control Limitations

Despite improvements, MQTT v5 has key limitations:

1. **QoS 0 Traffic Not Controlled**: No ACK → no feedback mechanism.
2. **No Security Awareness**: Flow control does not consider authentication or trust level.
3. **Static Thresholds**: Not adaptive to dynamic workloads.
4. **Wildcard Blindness**: Does not account for fan-out amplification.

These limitations are explicitly highlighted in recent literature.

2.6. Interaction Between QoS, Flow Control, and Security

A critical observation:

- QoS → increases reliability but adds overhead.
- Security → increases processing cost.
- Flow control → regulates traffic but lacks context awareness.

This interaction can be expressed as: $\boldsymbol{Load}total = \boldsymbol{Load}traffic + \boldsymbol{Load}security + \boldsymbol{Load}QoS$

Where:

- $\boldsymbol{Load}security$: TLS, authentication, ACL checks.
- $\boldsymbol{Load}QoS$: retransmissions, acknowledgments.

When combined: $\boldsymbol{Load}total > \boldsymbol{Capacity}br \Rightarrow$ Backpressure

2.7. Summary of Background and Key Insights

From this background, several critical insights emerge:

1. MQTT's lightweight design enables scalability but introduces **system-level vulnerabilities under load**.
2. QoS mechanisms improve reliability but contribute to **traffic amplification and latency**.
3. MQTT v5 flow control provides partial solutions but lacks **adaptability and context awareness**.
4. Wildcard subscriptions significantly increase **routing complexity and security risks**.
5. Backpressure is a **system-level phenomenon** influenced by multiple interacting factors.

These observations motivate the need for **integrated and adaptive approaches**, which are explored in the following sections.

3. Security Mechanisms in MQTT Systems

3.1. Overview of Security Requirements in MQTT-Based IoT

Security in MQTT-based IoT systems must ensure the classical **Confidentiality, Integrity, and Availability (CIA)** triad, in addition to authentication, authorization, and accountability. Due to the large-scale, heterogeneous, and resource-constrained nature of IoT environments, implementing these security requirements presents significant challenges [21,22].

Unlike traditional enterprise protocols, MQTT does not enforce built-in security at the protocol level. Instead, it relies on external mechanisms such as Transport Layer Security (TLS), credential-based authentication, and broker-enforced access control policies [5]. As highlighted in this research review, this design introduces a **trade-off between security enforcement and system performance**, particularly under high-load conditions.

3.2. Authentication Mechanisms in MQTT

Authentication ensures that only legitimate clients can connect to the broker.

 doi:10.20944/preprints202604.0222.v1

3.2.1. Username and Password Authentication

The simplest and most widely used mechanism is **username/password authentication**, where credentials are transmitted during the CONNECT phase [23].

Advantages:

- Lightweight and easy to implement.
- Suitable for constrained devices.

Limitations:

- Vulnerable to brute-force and replay attacks if not encrypted.
- Requires secure transport (TLS).

3.2.2. Certificate-Based Authentication (TLS/X.509)

TLS with **X.509 certificates** provides strong mutual authentication between client and broker.

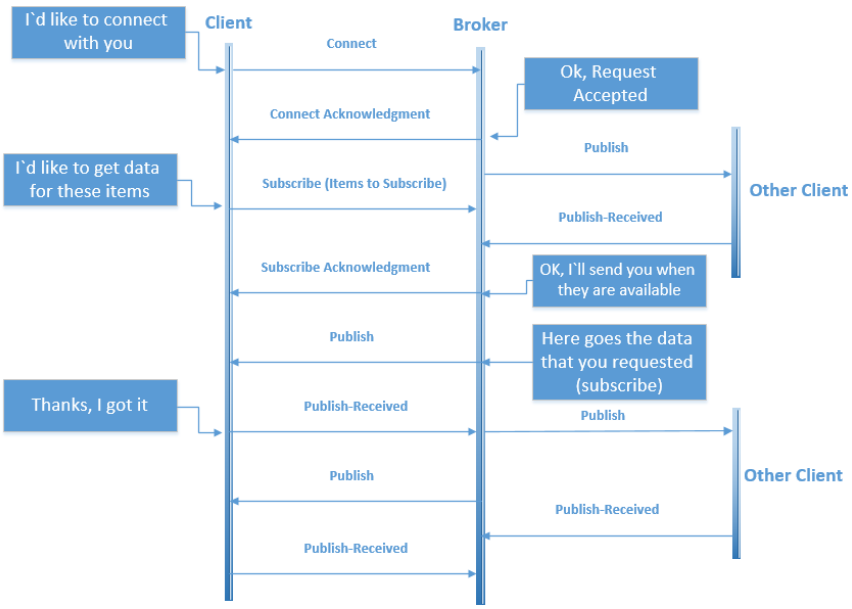

Figure 5. TLS-based authentication process in MQTT systems.

Advantages:

- Strong identity verification.
- Protection against eavesdropping and MitM attacks.

Limitations:

- High computational overhead.
- Increased latency due to handshake.

Recent studies show that TLS handshake latency can significantly impact IoT performance, especially during reconnection storms or large-scale deployments [24].

3.2.3. Token-Based Authentication (OAuth, JWT)

Modern MQTT systems increasingly adopt **token-based authentication**, such as OAuth 2.0 or JSON Web Tokens (JWT).

Advantages:

- Scalable and flexible.
- Suitable for cloud-integrated IoT.

Limitations:

- Token validation overhead.

 doi:10.20944/preprints202604.0222.v1

- Dependency on external identity providers.

3.2.4. Authentication Overhead Model

Authentication contributes to system load: $\boldsymbol{L}auth = \boldsymbol{C}crypto + \boldsymbol{C}handshake + \boldsymbol{C}verification$
Where:

- $\boldsymbol{C}crypto$: encryption/decryption cost.
- $\boldsymbol{C}handshake$: connection setup cost.
- $\boldsymbol{C}verification$: credential validation.

As demonstrated in my experimental results, authentication overhead increases latency and contributes to backpressure under high-load conditions.

3.3. Authorization and Access Control

Authorization determines whether an authenticated client can publish or subscribe to specific topics.

3.3.1. Access Control Lists (ACLs)

ACLs are the most common authorization mechanism in MQTT brokers [25].
Example: *user1 → publish → home/temperature*
user2 → subscribe → factory/#

Advantages:

- Fine-grained control.
- Easy integration with brokers.

Limitations:

- High evaluation overhead.
- Complexity in large-scale systems.

3.3.2. Role-Based Access Control (RBAC)

RBAC assigns permissions based on roles rather than individual users.
Advantages:

- Scalable for large systems.
- Easier policy management.

Limitations:

- Less flexible than attribute-based models.

3.3.3. Attribute-Based Access Control (ABAC)

ABAC evaluates policies based on attributes (user, device, context).
Advantages:

- Highly flexible and dynamic.
- Context-aware decisions.

Limitations:

- High computational complexity.
- Difficult to implement in constrained environments.

3.3.4 Authorization Overhead
Authorization cost per message: $\boldsymbol{L}authz = (\boldsymbol{N}rules, \boldsymbol{C}match)$
Where:

- $\boldsymbol{N}rules$: number of ACL rules.
- $\boldsymbol{C}match$: complexity of topic matching.

 doi:10.20944/preprints202604.0222.v1

In wildcard-intensive systems, C_{match} increases significantly, amplifying broker load.

3.4. Encryption and Data Protection

Encryption ensures confidentiality and integrity of data.

3.4.1. Transport-Level Security (TLS)

TLS is the most common mechanism used in MQTT.

Security Benefits:

- Confidentiality.
- Integrity.
- Authentication.

Performance Impact:

- Increased packet size.
- CPU overhead.
- Latency increase.

3.4.2. Payload Encryption

In some cases, end-to-end encryption is applied at the application level.

Advantages:

- Protects data beyond broker.
- Enhances privacy.

Limitations:

- Additional processing overhead.
- Key management complexity.

3.5. Security Threats in MQTT Systems

MQTT systems are exposed to several threats:

3.5.1. Unauthorized Access

Weak authentication or misconfigured ACLs may allow unauthorized clients to:

- Publish malicious data.
- Subscribe to sensitive topics.

3.5.2. Wildcard Exploitation

Wildcard subscriptions (e.g., #) can:

- Expose large data sets.
- Enable data exfiltration.

As highlighted in this research review, wildcard misuse significantly increases both **security risk and system load**.

3.5.3. Denial-of-Service (DoS) Attacks

Attackers can:

- Flood broker with messages.
- Exploit QoS retransmissions.
- Trigger backpressure.

 doi:10.20944/preprints202604.0222.v1

3.5.4. Slow Subscriber Attack

A subscriber intentionally delays consumption, causing:

- Queue buildup.
- Increased latency.
- System instability.

3.6. Security vs Performance Trade-Off

A fundamental challenge in MQTT systems is balancing security and performance.

3.6.1. Combined System Load Model

$$\boldsymbol{L}total = \boldsymbol{L}traffic + \boldsymbol{L}security + \boldsymbol{L}routing$$

Where:

- $\boldsymbol{L}security = \boldsymbol{L}auth + \boldsymbol{L}authz + \boldsymbol{L}encryption$
- $\boldsymbol{L}routing$: wildcard matching cost

If:

$$\boldsymbol{L}total > \boldsymbol{C}apacitybroker \Rightarrow \text{Backpressure and instability}$$

3.6.2. Comparative Analysis

Table 2. Security mechanisms vs performance trade-offs.

Mechanism	Security Strength	Performance Impact	Scalability
Username/Password	Low–Medium	Low	High
TLS Certificates	High	High	Medium
OAuth/JWT	High	Medium	High
ACL	Medium	Medium	Medium
ABAC	High	High	Low

3.7. Key Insights and Limitations of Existing Approaches

From the literature and my experimental results findings:

1. Security mechanisms significantly increase **latency and processing overhead**.
2. Authorization complexity grows with **wildcard usage and topic hierarchy size**.
3. Existing solutions treat security and performance **independently**.
4. No unified framework integrates:
   - Authentication.
   - Authorization.
   - Flow control.
   - Wildcard behavior.

This reinforces the central research gap identified in Section 1.

3.8. Summary

This section demonstrates that security mechanisms are essential for protecting MQTT systems but introduce significant overhead that directly impacts system performance. Authentication, authorization, and encryption mechanisms contribute to increased latency, computational cost, and traffic volume, particularly in large-scale IoT environments. Furthermore, wildcard subscriptions and access control policies interact in complex ways, amplifying both security risks and performance degradation.

These findings highlight the need for **integrated, security-aware flow-control mechanisms**, which are explored in the next section.

 doi:10.20944/preprints202604.0222.v1

4. Backpressure and Flow-Control Mechanisms in MQTT-Based IoT Systems

4.1. Overview of Backpressure in MQTT Systems

4.1.1. Default MQTT Publishing Behavior (Baseline Scenario)

As illustrated in Figure 6, the default MQTT publishing model follows a continuous loop where messages are generated, formatted, and transmitted without considering system load or broker capacity. This behavior can lead to excessive message rates, especially under time-driven publishing conditions, thereby contributing to backpressure and system congestion.

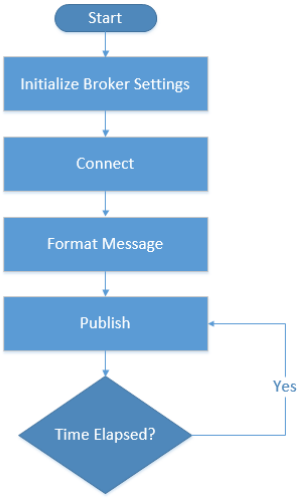

Figure 6. Default MQTT publishing workflow without flow control, illustrating continuous message transmission based on time-driven conditions.

This figure:

- Shows how MQTT works without flow control.
- Represents the problem (cause of backpressure).
- Serves as a reference model to compare the adaptive algorithm.
- Scenario X: Default publishing (Figure 6).
- Scenario Y: Adaptive flow control (Algorithm 1 + Figure 9).

Backpressure is a fundamental system-level phenomenon that arises when the rate of message production exceeds the processing and delivery capacity of the MQTT broker or its subscribers. In large-scale IoT deployments, this condition is common due to bursty traffic, heterogeneous device capabilities, and dynamic network conditions.

Formally, backpressure occurs when: $(t) > \mu service(t)$

Where:

- (t): incoming message rate.
- $\mu serv(t)$: broker service rate.

Under this condition, message queues grow over time: $(t+1) = (t) + \lambda i(t) - \mu service(t)$

If sustained: $(t) \rightarrow \infty \Rightarrow$ System instability

My experimental results experimental findings confirm that under high publishing rates, MQTT brokers (e.g., Mosquitto) experience rapid queue growth, increased latency, and message loss, especially when security and wildcard mechanisms are active [5].

4.2. Sources of Backpressure in MQTT Systems

Backpressure in MQTT is not caused by a single factor but by the interaction of multiple system components.

 doi:10.20944/preprints202604.0222.v1

4.2.1. High Publish Rate and Traffic Bursts

- Rapid data generation from IoT devices.
- Event-driven spikes (e.g., alarms, system updates).

4.2.2. Slow Subscribers (Receiver-Side Bottleneck)

- Limited processing capability.
- Network delays.
- Offline or intermittent clients.

4.2.3. Security Overhead

- TLS encryption/decryption.
- Authentication and authorization checks.

As shown in this research review, security mechanisms amplify congestion by increasing processing time and message size.

4.2.4. Wildcard Subscription Fan-Out

- Single message → multiple subscribers.
- Increased routing complexity.

4.2.5. QoS Overhead

Retransmissions (QoS 1/2)
Acknowledgment traffic

4.3. Backpressure Propagation Model

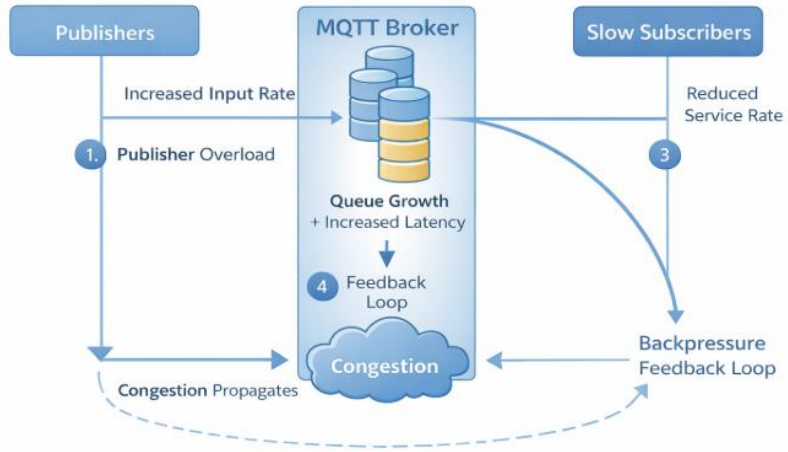

Figure 7. Backpressure propagation in MQTT systems.

Backpressure propagates through the system as follows:

1. Publisher overload → increased input rate.
2. Broker queue growth → increased latency.
3. Subscriber delay → reduced service rate.
4. Feedback loop → system-wide congestion.

This creates a **positive feedback loop**, where congestion reinforces itself [26].

4.4. Flow-Control Mechanisms in MQTT

Flow control aims to regulate message transmission to maintain system stability.

 doi:10.20944/preprints202604.0222.v1

4.4.1. TCP-Level Flow Control

MQTT relies on TCP for transport-level congestion control [27].

Limitations:

- Operates below application layer.
- Cannot control publish rate directly.
- Ineffective for multicast (publish/subscribe) systems.

Studies show TCP is insufficient for application-level overload in IoT messaging systems [28].

4.4.2. Broker-Level Flow Control

Common broker strategies include:

- Message queue limits.
- Dropping messages.
- Rate limiting.

Example (Mosquitto):

- Max inflight messages.
- Max queued messages.

Limitations:

- Reactive (not proactive).
- May cause message loss.
- No adaptation to system dynamics.

4.4.3. MQTT v5 Flow-Control Mechanisms

MQTT v5 introduces protocol-level control features.

(a) Receive Maximum $\boldsymbol{Ninflight} \leq \boldsymbol{Rmax}$

Limits the number of unacknowledged QoS 1/2 messages.

(b) Send Quota Model $\boldsymbol{Q}u(t+1) = \boldsymbol{Quota}(t) - 1 + \boldsymbol{ACK}(t)$

Transmission stops when: $\boldsymbol{Quota} = 0$

(c) Limitations of MQTT v5 Flow Control

Despite improvements:

1. No control for **QoS 0 traffic**.
2. No awareness of **security overhead**.
3. No adaptation to **dynamic workloads**.
4. No handling of **wildcard fan-out effects**.

These limitations are explicitly identified in this research review work.

4.5. Adaptive Flow-Control Mechanisms

To overcome static limitations, adaptive mechanisms dynamically regulate the publish rate based on system feedback.

4.5.1. Adaptive Flow-Control Algorithm for Backpressure Mitigation

To clarify the operational logic of the proposed adaptive mechanism, Algorithm 1 summarizes the runtime procedure used to detect publishing-rate escalation and apply back-off delay when backpressure conditions are observed.

As shown in Algorithm 1, the adaptive mechanism relies on continuous monitoring of message arrival behavior and uses a threshold-based decision process to distinguish normal traffic from overload conditions, thereby reducing congestion and improving broker stability.

 doi:10.20944/preprints202604.0222.v1


```
Input:
    N                   ← number of observed messages
    current_arrival     ← current message arrival time
    avg_rate            ← average historical publishing rate
    threshold           ← acceptable publishing-rate threshold
    delay_factor        ← back-off delay value
Output:
    decision on whether to accept message immediately
    or apply delay before processing
Begin
1. Connect to MQTT broker
2. Record the arrival time of the first message
3. Repeat for N messages:
       a. Record message arrival times
       b. Count total received messages
4. Calculate the average publishing rate
5. Store the average publishing rate in a rate history array
6. For each new incoming message:
       a. Record the new message arrival time
       b. Calculate the new publishing rate
7. Compare the new publishing rate with the average publishing rate
8. If new publishing rate > average publishing rate + threshold then
       a. Calculate delay factor
       b. Apply back-off delay
       c. Accept delayed message
   Else
       a. Accept message immediately
   End If
9. Process the accepted message
End
```

Figure 8. Adaptive Flow-Control Algorithm for Backpressure Mitigation.

Explanation of Algorithm 1

Algorithm 1 presents an adaptive flow-control procedure designed to reduce backpressure in MQTT systems by regulating the message acceptance rate at runtime. The algorithm begins by connecting to the MQTT broker and monitoring the arrival times of an initial set of messages. These observations are used to calculate an average publishing rate, which serves as a baseline for normal traffic behavior.

After this initialization phase, the algorithm continuously records the arrival time of each new message and computes the current publishing rate. This current rate is then compared with the previously calculated average rate. If the new rate exceeds the acceptable range, the algorithm interprets the change as a possible overload condition and applies a back-off delay before accepting the message. Otherwise, the message is accepted and processed immediately.

The main purpose of this adaptive decision process is to prevent sudden bursts of traffic from overwhelming the broker or downstream subscribers. By introducing delay only when the incoming rate becomes excessive, the algorithm helps maintain system stability while preserving normal throughput under non-congested conditions. This makes the method suitable for MQTT environments where traffic patterns are dynamic and fixed-rate control is insufficient.

4.5.2. Backpressure-Driven Adaptive Flow Control (Core Concept)

This review proposes a **feedback-based adaptive mechanism**: $\boldsymbol{R}new = \boldsymbol{R}current \times (\boldsymbol{B}\ (t))$
Where:

- $\boldsymbol{R}new$: updated publish rate.
- (t): backpressure indicator.

Backpressure Indicator Function $(t) = w1 \cdot Laten(t) + w2 \cdot Queue(t) + w3 \cdot Loss(t)$
Control Law

$$R(t+1) = \begin{cases} R(t) \cdot \alpha, if\ B(t) > \theta \\ R(t) + \beta, otherwise \end{cases}$$

Where:

- $\alpha < 1$: reduction factor.
- $\beta > 0$: increase step.

 doi:10.20944/preprints202604.0222.v1

- θ: congestion threshold.

This creates a **negative feedback system** that stabilizes the broker.

4.5.3. Stability Interpretation

The system converges when: $\lambda in \approx \mu service$
My experimental results demonstrate that this approach:
Reduces queue growth
Stabilizes latency
Improves delivery success rate

4.5.4. Adaptive Flow-Control Algorithm for Backpressure Mitigation

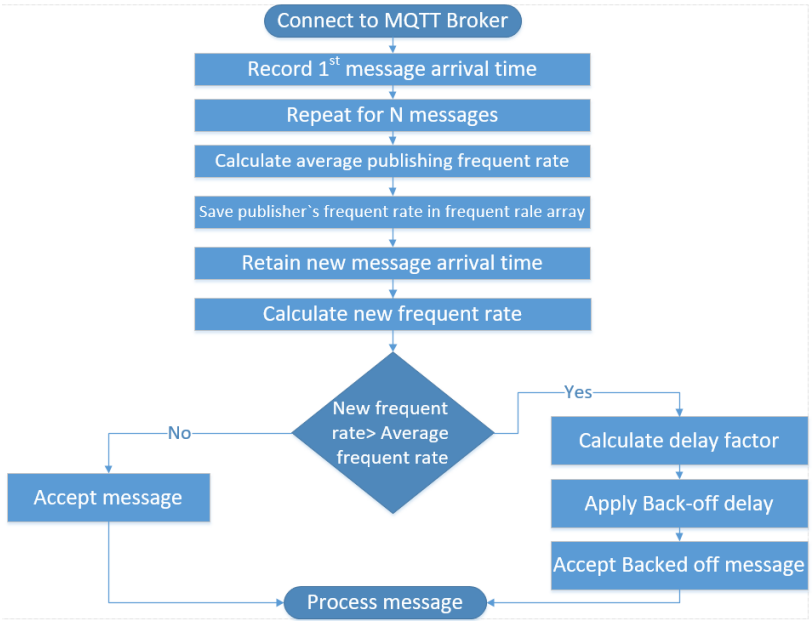

Figure 9. Adaptive backpressure-aware flow-control algorithm based on dynamic publishing rate adjustment.

As illustrated in Figure 9, the proposed adaptive flow-control mechanism dynamically monitors message arrival rates and compares them with the historical average to detect potential congestion. When the publishing rate exceeds the acceptable threshold, a back-off delay is applied to regulate traffic and prevent broker overload.

4.6. Comparison of Flow-Control Approaches

Table 3. Comparison of flow-control mechanisms.

Approach	Adaptivity	QoS Awareness	Security Awareness	Efficiency
TCP Control	Low	No	No	Low
Broker Limits	Low	Partial	No	Medium
MQTT v5	Medium	Yes (QoS1/2)	No	Medium
Adaptive Flow Control	High	Yes	Partial	High

4.7. Integration with Security and Wildcards

A key insight from this research review:

- Security increases processing delay → increases backpressure.
- Wildcards increase message fan-out → increases load.
- Flow control must consider both.

 doi:10.20944/preprints202604.0222.v1

This leads to an extended model: $\boldsymbol{Load}total = \boldsymbol{\lambda}traffic + \boldsymbol{\lambda}security + \boldsymbol{\lambda}wildcard$

Where:

$\boldsymbol{\lambda}security$: TLS + authentication overhead.

$\boldsymbol{\lambda}wildcard$: fan-out amplification.

If not controlled: $\boldsymbol{Load}total > \boldsymbol{Capacity}broker \Rightarrow$ Congestion

4.8. Limitations of Existing Flow-Control Mechanisms

From literature and this research review work:

1. Lack of cross-layer integration.
2. No security-aware flow control.
3. Limited handling of QoS 0 traffic.
4. No optimization for wildcard-heavy systems.
5. Absence of standardized adaptive frameworks.

4.9. Summary and Key Insights

This section demonstrates that:

- Backpressure is a multi-factor system problem.
- Existing solutions are fragmented and incomplete.
- MQTT v5 improves control but remains insufficient.
- Adaptive flow control provides a promising solution.
- Integration with security and wildcard management is essential.

These findings motivate the need for **holistic, adaptive, and security-aware frameworks**, which are analyzed in the next section.

5. Wildcard Subscription Behavior and Optimization in MQTT Systems

5.1. Overview of Wildcard Subscriptions in MQTT

MQTT supports hierarchical topic structures that enable flexible and scalable data dissemination. To simplify subscription management, MQTT introduces two wildcard operators:

- **Single-level wildcard (+)**: matches exactly one topic level.
- **Multi-level wildcard (#)**: matches all remaining levels.

These operators allow subscribers to receive messages from multiple topics using a single subscription, which is particularly useful in large-scale IoT systems such as smart buildings, industrial automation, and environmental monitoring.

However, while wildcard subscriptions enhance scalability and usability, they also introduce **significant performance and security challenges**, especially in high-load and heterogeneous environments. As emphasized in my research review work, wildcard-intensive systems can substantially increase broker workload and system instability.

5.2. Wildcard Matching Mechanism

MQTT brokers match incoming topics against subscription filters using pattern-matching algorithms [29].

5.2.1. Topic Matching Complexity

Let:

- $\boldsymbol{T}$: number of topics.
- $\boldsymbol{S}$: number of subscriptions.

The matching complexity can be approximated as: $\boldsymbol{C}match = (\boldsymbol{T} \times \boldsymbol{S})$

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 3 April 2026 doi:10.20944/preprints202604.0222.v1

In wildcard scenarios: $C_{wildcard} > C_{exact}$

Because:

- Each wildcard requires pattern evaluation.
- Multi-level wildcards increase matching depth.

5.2.2. Wildcard Matching Process

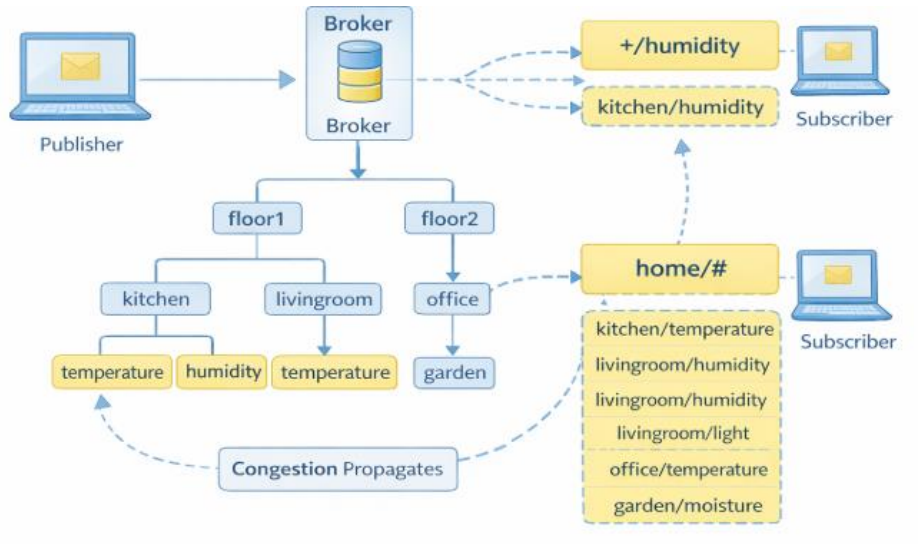

Figure 10. Topic matching with wildcard subscriptions in MQTT.

Brokers typically use:

- Trie-based structures.
- Hash tables.
- Tree traversal algorithms.

Wildcard subscriptions require **recursive matching**, increasing processing time and memory usage [30].

5.3. Performance Impact of Wildcard Subscriptions

Wildcard subscriptions significantly affect system performance in multiple ways [31].

5.3.1. Routing Overhead

A single published message may be delivered to multiple subscribers: $F_{anout} = |\{Subscribers_{match}\}|$

Total delivery cost: $C_{delivery} = F_{anout} \times C_{forward}$

Where:

- $C_{forward}$: cost per message forwarding.

This research review results show that multi-level wildcards (#) produce the highest routing overhead and CPU utilization.

5.3.2. CPU Utilization

Wildcard matching increases:

- String comparison operations.
- Tree traversal depth.
- Memory access.

Resulting in: $CPU_{load} \propto C_{wildcard} \times F_{anout}$

 doi:10.20944/preprints202604.0222.v1

5.3.3. Latency Increase

Latency grows with:

- Matching complexity.
- Queue size.
- Fan-out delivery. $\boldsymbol{Latency} = f\ (\boldsymbol{C}match, \boldsymbol{Q}\ (t), \boldsymbol{Fanout})$

The research review confirms that wildcard-heavy configurations lead to **significant latency growth under high load**.

5.4. Security Risks of Wildcard Subscriptions

Wildcard subscriptions introduce critical security vulnerabilities [32].

5.4.1. Unauthorized Data Access

Broad subscriptions such as: #

home/#

May unintentionally expose:

- Sensitive data.
- System-wide telemetry.

If access control policies are weak or misconfigured.

5.4.2. Data Exfiltration Attacks

Attackers can:

- Subscribe to #.
- Collect all broker messages.

This creates a **data harvesting attack vector** [33].

5.4.3. Traffic Amplification

Wildcard subscriptions increase traffic volume: $\boldsymbol{Traffic}amplified = \boldsymbol{Messages} \times \boldsymbol{Fanout}$

This can lead to:

- Broker overload.
- Denial-of-Service (DoS) conditions.

5.4.4. Interaction with Access Control

Authorization checks become more complex: $\boldsymbol{C}authz = \boldsymbol{f}\ (\boldsymbol{N}rules, \boldsymbol{C}wildcard)$

Where:

- More wildcard rules → higher evaluation cost.

This research review highlights that wildcard misconfiguration significantly increases both **security risk and processing overhead**.

5.5. Mathematical Modeling of Wildcard Overlap

A key problem in wildcard subscriptions is **overlapping topic coverage**, which leads to redundant message delivery.

5.5.1. Misrouting Probability Model

Let:

- $\boldsymbol{P}overlap$: probability of overlap.
- $\boldsymbol{N}topics$: number of topics.
- $\boldsymbol{N}wildcards$: number of wildcard subscriptions.

 doi:10.20944/preprints202604.0222.v1

$$P_{overlap} = \frac{N_{overlapping\_matches}}{N_{topics}}$$

Higher overlap leads to:

- Duplicate message delivery.
- Increased bandwidth usage.

My own experimental results provide a detailed analysis of **misrouting probability and overlap reduction strategies**.

5.5.2. Optimization Objective

Minimize: $\boldsymbol{Cost} = \boldsymbol{\alpha} \cdot \boldsymbol{C}match + \beta \cdot \boldsymbol{Fanout} + \gamma \cdot \boldsymbol{P}overlap$
Where: α, β, γ: weighting factors.

5.6. Optimization Techniques for Wildcard Management

5.6.1. Precise Topic Design

Replace broad wildcards with structured topics:
(False) *home/#*
(True) *home/livingroom/temperature*
Benefits:

- Reduced overlap.
- Improved security.
- Lower routing cost.

5.6.2. Subscription Partitioning

Divide large wildcard subscriptions into smaller segments: *home/# \rightarrow \{home/floor1/+, home/floor2/+\}*

5.6.3. Broker-Side Optimization

Techniques include:

- Trie-based indexing.
- Caching subscription matches.
- Precomputed routing tables.

5.6.4. Adaptive Wildcard Control

Integrate wildcard behavior into flow control: $\boldsymbol{Rate}adjusted = \boldsymbol{f}\ (\boldsymbol{Fanout}, \boldsymbol{CPU}load)$
This ensures:

- Load-aware subscription handling.
- Reduced congestion.

5.6.5. Access Control Enhancement

Restrict wildcard usage:

- Limit # access.
- Apply strict ACL policies.
- Role-based filtering.

5.7. Integration with Flow Control and Security

Wildcard behavior interacts with other system components:
Combined Load Model: $\boldsymbol{Load}total = \boldsymbol{\lambda}traffic + \boldsymbol{\lambda}security + \boldsymbol{\lambda}wildcard$

 doi:10.20944/preprints202604.0222.v1

Where: $\lambda wildcard = Fanout \times MatchingCost$

System Instability Condition: $Loadtotal > Capacitybroker \Rightarrow$ **Backpressure**

This confirms that wildcard subscriptions are a **key contributor to system instability**, especially when combined with security overhead.

5.8. Comparative Analysis of Wildcard Strategies

Table 4. Comparison of wildcard management strategies.

Strategy	Performance	Security	Complexity
Broad wildcard (#)	Low	Low	Low
Limited wildcard (+)	Medium	Medium	Medium
Precise topics	High	High	High
Adaptive wildcard control	High	High	High

5.9. Key Insights and Limitations

From the literature and my own experimental findings:

1. Wildcards significantly increase:
   - Routing complexity.
   - CPU utilization.
   - Latency.
2. Wildcard misuse creates:
   - Security vulnerabilities.
   - Data exposure risks.
3. Existing systems lack:
   - Wildcard-aware flow control.
   - Integrated security policies.
   - Optimization frameworks.

5.10. Summary

This section demonstrates that wildcard subscriptions, while essential for scalability, introduce substantial performance and security challenges in MQTT systems. Their impact is amplified in large-scale IoT deployments, where high message rates, complex topic hierarchies, and dynamic workloads are common.

The analysis shows that:

- Wildcard subscriptions increase routing complexity and system load.
- They introduce critical security risks when improperly controlled.
- Existing solutions fail to integrate wildcard behavior with flow control and security mechanisms.

These findings reinforce the need for **integrated, adaptive, and security-aware optimization strategies**, which are discussed in the next section.

6. Integrated Analysis and Literature Gap Discussion

6.1. Overview of Integrated System Behavior

The preceding sections have independently examined three fundamental dimensions of MQTT-based IoT systems:

- **Security mechanisms** (authentication, authorization, encryption).
- **Flow control and backpressure management**.
- **Wildcard subscription behavior and routing complexity**.

 doi:10.20944/preprints202604.0222.v1

However, in real-world deployments, these components do not operate independently. Instead, they form a **tightly coupled system**, where each dimension directly influences the others.

Your dissertation and review work explicitly demonstrate that system instability emerges from **the interaction of these factors rather than from any single mechanism in isolation**. This observation motivates the need for an integrated analysis.

6.2. Cross-Layer Interaction Model

To understand the system holistically, we define a combined load model:

$$\boldsymbol{Load}total = \boldsymbol{Load}traffic + \boldsymbol{Load}security + \boldsymbol{Load}wildcard$$

Where:

- $\boldsymbol{Load}traffic$: base message rate.
- $\boldsymbol{Load}security$: authentication + encryption + authorization overhead.
- $\boldsymbol{Load}wildcard$: fan-out and matching complexity.

System Stability Condition $\boldsymbol{Load}total \leq \boldsymbol{Capacity}broker$

If: $\boldsymbol{Load}total > \boldsymbol{Capacity}broker \Rightarrow$ **Backpressure → Latency → Message Loss → System Failure**

This formulation reflects the empirical findings in your experiments, where increasing authentication overhead and wildcard intensity led to rapid degradation in system performance.

6.3. Interaction Between Security and Flow Control

Security mechanisms significantly influence system performance:

- **TLS encryption** increases packet size and processing cost.
- **Authentication** adds connection latency.
- **Authorization** introduces per-message evaluation overhead.

These factors increase: $\mu_{service}^{-1}$ ⇒ **Slower processing rate**

Thus: $\boldsymbol{\lambda}in > \boldsymbol{\mu}service \Rightarrow$ **Backpressure**

Recent studies confirm that security overhead can degrade MQTT throughput and increase latency under load [34,35].

Key Insight

Security mechanisms, while essential, act as **backpressure amplifiers** when not integrated with flow-control strategies.

6.4. Interaction Between Wildcards and Flow Control

Wildcard subscriptions directly impact traffic dynamics: $\boldsymbol{Traffic}effective = \boldsymbol{\lambda}in \times \boldsymbol{Fanout}$

Where: $\boldsymbol{F}anout$: number of matched subscribers.

This leads to:

- Increased routing cost.
- Increased broker load.
- Higher probability of congestion.

Your experimental analysis shows that multi-level wildcards (#) significantly increase CPU utilization and latency, especially under high-load conditions.

Key Insight: Wildcard subscriptions act as **traffic amplifiers**, accelerating the onset of backpressure.

6.5. Interaction Between Security and Wildcards

The interaction between security and wildcard behavior is particularly critical:

- Wildcards increase authorization complexity.
- ACL evaluation becomes more expensive.
- Misconfigurations increase security risks.

 doi:10.20944/preprints202604.0222.v1

Example: subscribe: #

→ Requires checking access for **all topics**, increasing overhead: $C_{authz} \propto C_{wildcard}$

Your review highlights that wildcard misuse can lead to unauthorized data access and traffic amplification.

Key Insight

Wildcards and security mechanisms together create a **complex, high-cost decision space** at the broker.

6.6. Combined Effect: System-Level Bottleneck

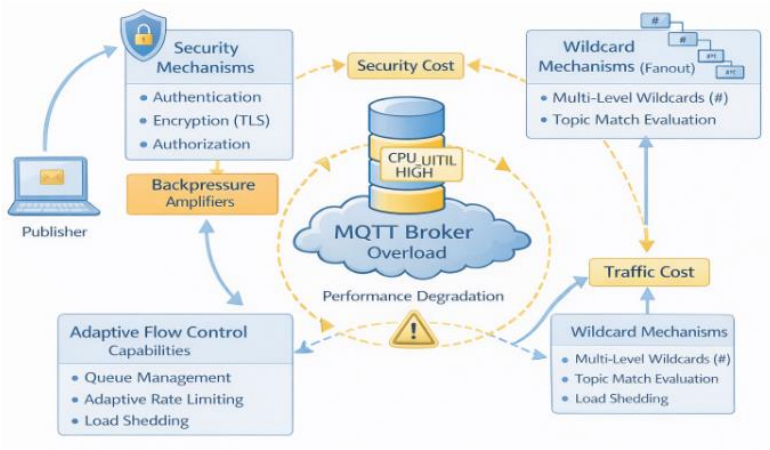

Figure 11. Integrated interaction between security, flow control, and wildcard mechanisms.

The combined interaction leads to:

1. Increased processing overhead.
2. Increased routing complexity.
3. Reduced service rate.
4. Positive feedback loop of congestion.

6.7. Critical Review of Existing Literature

Existing research can be categorized into three main groups:

6.7.1. Security-Focused Studies

Focus:

- Authentication and encryption.
- Access control.

Limitations:

- Ignore performance impact.
- No integration with flow control.

Examples:

- Sicari et al. (2015) [36].
- Yang et al. (2021) [37].

6.7.2. Flow-Control and Performance Studies

Focus:

- Backpressure mitigation.
- Rate control.

 doi:10.20944/preprints202604.0222.v1

Limitations:

- Ignore security overhead.
- No wildcard consideration.

Examples:

- Liu et al. (2021) [38].
- Wu et al. (2025) [39].

6.7.3. Wildcard and Routing Studies

Focus:

- Topic matching optimization.
- Routing efficiency.

Limitations:

- Ignore security implications.
- No congestion analysis.

Examples: Mishra et al. (2020) [40].

6.8. Literature Gap Matrix

Table 5. Literature gap analysis.

Dimension	Security Studies	Flow-Control Studies	Wildcard Studies	Integrated Approaches
Authentication	✓	✗	✗	Partial
Authorization	✓	✗	✓	Partial
Backpressure	✗	✓	✗	✗
Adaptive Flow Control	✗	✓	✗	✗
Wildcard Optimization	✗	✗	✓	✗
Cross-Layer Integration	✗	✗	✗	✗

6.9. Identified Research Gap

From the analysis above, a clear and critical research gap emerges:

Existing research lacks a unified, cross-layer framework that integrates security mechanisms, adaptive flow control, and wildcard-aware optimization in MQTT-based IoT systems.

Specifically:

1. No security-aware flow control mechanisms.
2. No wildcard-aware congestion control.
3. No joint evaluation under realistic workloads.
4. Lack of adaptive, context-aware broker behavior.

6.10. Positioning of This Review

This review addresses the identified gap by:

- Providing a **holistic synthesis** of MQTT system behavior.
- Highlighting **interdependencies between core mechanisms**.
- Identifying **limitations in existing approaches**.
- Establishing the need for:
  - Adaptive flow control.

 doi:10.20944/preprints202604.0222.v1

- Security-aware optimization.
- Wildcard-aware routing strategies.

The research review contributes to this gap by proposing:

- Backpressure-driven adaptive flow control.
- Experimental validation under security and wildcard conditions.
- Integrated performance analysis.

6.11. Key Insights from Integrated Analysis

The integrated analysis reveals:

1. MQTT system performance is governed by **multi-factor interactions**.
2. Security mechanisms are **not performance-neutral**.
3. Wildcards significantly amplify both:
   - Traffic.
   - Security risk.
4. Backpressure is a system-wide emergent behavior.
5. Current solutions are fragmented and insufficient.

6.12. Summary

This section demonstrates that MQTT-based IoT systems must be analyzed as **integrated, cross-layer systems**, where security, flow control, and wildcard behavior interact dynamically. The literature review reveals that most existing approaches focus on individual aspects in isolation, leading to incomplete solutions that fail under real-world conditions.

The identified gap highlights the urgent need for **adaptive, security-aware, and wildcard-aware frameworks**, which are essential for achieving scalable, reliable, and secure IoT communication.

7. Future Research Directions

7.1. Overview

The integrated analysis presented in Section 6 highlights that current MQTT-based IoT systems suffer from fragmented solutions that address security, flow control, and wildcard behavior independently. As demonstrated in my research review and supporting studies, real-world IoT deployments require **holistic, adaptive, and cross-layer approaches** to ensure scalability, reliability, and security under dynamic conditions.

This section outlines key future research directions that aim to bridge the identified gaps and advance the state of the art in MQTT-based IoT systems.

7.2. Security-Aware Adaptive Flow Control

7.2.1. Motivation

Existing flow-control mechanisms (e.g., MQTT v5 Receive Maximum) do not consider security overhead. However, authentication, encryption, and authorization directly affect system performance.

7.2.2. Research Direction

Develop **security-aware adaptive flow-control algorithms** that dynamically adjust message rates based on both congestion indicators and security context.

Proposed Model $\boldsymbol{R}adaptive = \boldsymbol{f}\ (\boldsymbol{B}\ (t), \boldsymbol{S}\ (t))$

Where:

- (t): backpressure indicator (latency, queue size, loss).

 doi:10.20944/preprints202604.0222.v1

- (t): security cost (authentication delay, encryption overhead).

Expected Contributions

- Reduced congestion under secure communication.
- Improved QoS reliability.
- Balanced trade-off between security and performance.

This research review already demonstrates the effectiveness of adaptive flow control under varying conditions, providing a strong foundation for this direction.

7.3. Wildcard-Aware Flow Control and Routing Optimization

7.3.1. Motivation

Wildcard subscriptions significantly increase message fan-out and routing complexity, yet current flow-control mechanisms ignore this factor.

7.3.2. Research Direction

Design wildcard-aware congestion control mechanisms that incorporate subscription patterns into traffic regulation.

Proposed Model $R_{\text{adjusted}} = \frac{R_{\text{base}}}{1+ \text{Fanout} \cdot W_{\text{factor}}}$

Where:

- $Fanout$: number of matched subscribers.
- $Wfactor$: wildcard complexity weight.

Expected Contributions

- Reduced routing overhead.
- Controlled traffic amplification.
- Improved scalability in large topic hierarchies.

7.4. Cross-Layer Optimization Frameworks

7.4.1. Motivation

Current MQTT systems lack coordination between:

- Application layer (publish rate).
- Security layer (authentication, ACL).
- Transport layer (TCP).

7.4.2. Research Direction

Develop **cross-layer optimization frameworks** that jointly manage:

- Traffic generation.
- Security enforcement.
- Routing decisions.

Integrated Optimization Objective $\min Cost = \alpha \cdot Latency + \beta \cdot Loss + \gamma \cdot SecurityRisk$
Subject to: $Loadtotal \leq Capacitybroker$

Expected Contributions

- Holistic system optimization.
- Improved stability under dynamic workloads.
- Reduced performance-security trade-offs.

7.5. Intelligent and AI-Driven MQTT Management

 doi:10.20944/preprints202604.0222.v1

7.5.1. Motivation

Static thresholds and rule-based systems cannot adapt to dynamic IoT environments.

7.5.2. Research Direction

Apply **machine learning (ML) and artificial intelligence (AI)** techniques for:

- Congestion prediction.
- Anomaly detection.
- Adaptive rate control.

Example Approach $(t + 1) = \boldsymbol{ML\_Model}\ (\boldsymbol{Features}t)$

Where: Features include:

- Latency.
- Queue size.
- Security events.
- Wildcard usage.

Expected Contributions

- Predictive congestion avoidance.
- Self-optimizing brokers.
- Improved resilience against attacks.

Recent works highlight the growing role of AI in IoT optimization and security management [41,42].

7.6. Advanced Security Mechanisms for MQTT

7.6.1. Motivation

Traditional security mechanisms introduce overhead and are not optimized for IoT constraints.

7.6.2. Research Direction

Explore lightweight and scalable security solutions:

- Lightweight cryptography.
- Zero-trust architectures.
- Blockchain-based authentication.

Expected Contributions

- Reduced computational overhead.
- Enhanced trust management.
- Improved scalability.

Recent studies emphasize the need for lightweight and decentralized security models in IoT systems [43].

7.7. Wildcard Optimization and Topic Design Automation

7.7.1. Motivation

Manual topic design leads to inefficiencies and security risks.

7.7.2. Research Direction

Develop **automated topic optimization tools** that:

- Minimize wildcard overlap.
- Reduce routing complexity.

 doi:10.20944/preprints202604.0222.v1

- Enhance security.

Optimization Objective min ($\boldsymbol{C}match + \boldsymbol{F}anout + \boldsymbol{P}overlap$)

Expected Contributions

- Reduced broker load.
- Improved routing efficiency.
- Enhanced access control.

This research review already introduces mathematical modeling for wildcard optimization, which can be extended into automated systems.

7.8. Standardized Benchmarking and Evaluation Frameworks

7.8.1. Motivation

Current research lacks standardized evaluation methodologies.

7.8.2. Research Direction

Develop **benchmarking frameworks** for MQTT systems that include:

- Security metrics.
- Performance metrics.
- Scalability analysis.

Key Metrics

- Latency.
- Throughput.
- Message loss rate.
- Authentication delay.
- CPU utilization.

Expected Contributions

- Reproducible experiments.
- Fair comparison of solutions.
- Improved research quality.

7.9. Integration with Edge and Fog Computing

7.9.1. Motivation

Centralized brokers become bottlenecks in large-scale systems.

7.9.2. Research Direction

Investigate **distributed MQTT architectures**:

- Edge-based brokers.
- Fog computing integration.
- Hierarchical messaging systems.

Expected Contributions

- Reduced latency.
- Improved scalability.
- Enhanced fault tolerance.

 doi:10.20944/preprints202604.0222.v1

7.10. Toward Self-Adaptive and Autonomous MQTT Systems

7.10.1. Vision

Future MQTT systems should be:

- Self-monitoring.
- Self-optimizing.
- Self-healing.

Research Direction

Develop **autonomous MQTT systems** that:

- Detect congestion.
- Adapt flow control.
- Adjust security policies dynamically.

Expected Contributions

- Fully adaptive IoT communication.
- Minimal human intervention.
- High resilience under dynamic conditions.

7.11. Summary of Future Directions

The future of MQTT-based IoT systems lies in:

1. Security-aware adaptive flow control.
2. Wildcard-aware optimization strategies.
3. Cross-layer system design.
4. AI-driven intelligent management.
5. Lightweight and scalable security mechanisms.
6. Standardized benchmarking frameworks.
7. Distributed and edge-based architectures.

These directions directly address the limitations identified in this review and provide a roadmap for developing **next-generation MQTT systems** that are secure, scalable, and resilient.

8. Conclusions

8.1. Summary of the Review

This review has presented a comprehensive and critical analysis of the interplay between **security mechanisms, backpressure and flow-control strategies, and wildcard subscription behavior** in MQTT-based IoT systems. While MQTT remains a widely adopted protocol due to its lightweight design and scalability, the analysis demonstrates that its performance and reliability degrade significantly under realistic deployment conditions characterized by high traffic volumes, complex subscription patterns, and stringent security requirements.

The review systematically examined three core dimensions. First, it analyzed **security mechanisms**, including authentication, authorization, and encryption, highlighting their essential role in protecting IoT systems while simultaneously introducing computational overhead and latency. Second, it investigated **backpressure and flow-control mechanisms**, showing that existing solutions—such as TCP-based control, broker-level limits, and MQTT v5 features—are insufficient to handle dynamic workloads and heterogeneous environments. Third, it explored **wildcard subscription behavior**, demonstrating that while wildcards improve scalability, they significantly increase routing complexity, traffic amplification, and security exposure.

8.2. Key Findings

The integrated analysis across these dimensions reveals several important findings:

 doi:10.20944/preprints202604.0222.v1

1. **Security and performance are inherently coupled**

Security mechanisms, including TLS and access control enforcement, introduce non-negligible overhead that directly contributes to congestion and backpressure in MQTT systems.

2. **Backpressure is a system-level emergent phenomenon**

It arises from the interaction between message rates, broker capacity, subscriber behavior, and security overhead, rather than from a single isolated factor.

3. **Wildcard subscriptions act as amplifiers of both load and risk**

Multi-level wildcard patterns significantly increase message fan-out, CPU utilization, and latency, while also expanding the attack surface.

4. **Existing solutions are fragmented and insufficient**

Most prior work addresses security, flow control, or routing independently, without considering their combined effects under realistic IoT workloads.

These findings are strongly supported by the experimental and analytical results presented in my own experiments, which demonstrate that MQTT systems experience rapid degradation under high load, while adaptive mechanisms can significantly improve stability and reliability.

8.3. Contributions of This Review

This review contributes to the literature in several important ways:

- **Comprehensive Synthesis:**

It provides a unified analysis of MQTT security, backpressure, and wildcard behavior, bridging previously disconnected research areas.

- **Critical Gap Identification:**

It identifies the absence of cross-layer, integrated approaches that jointly address security, flow control, and routing complexity.

- **System-Level Modeling:**

It introduces conceptual and mathematical models that explain how these components interact to influence system stability.

- **Research Roadmap:**

It outlines future research directions, including security-aware adaptive flow control, wildcard-aware optimization, and AI-driven system management.

8.4. Practical Implications

The findings of this review have significant implications for both researchers and practitioners:

- **For researchers:**

  There is a need to shift from isolated optimization techniques toward **holistic, system-level designs**.

- **For system designers:**

  MQTT deployments must consider:

  - Adaptive flow-control strategies
  - Controlled wildcard usage
  - Efficient and scalable security mechanisms

- **For IoT applications:**

Ensuring reliability and security requires **dynamic and context-aware system management**, particularly in large-scale environments such as smart cities, industrial IoT, and healthcare systems.

 doi:10.20944/preprints202604.0222.v1

8.5. Toward Next-Generation MQTT Systems

The analysis presented in this review supports a clear vision for the evolution of MQTT-based systems:

Future MQTT architectures must transition from static, protocol-driven designs to ***adaptive, intelligent, and cross-layer optimized systems****.*

Such systems should:

- Dynamically regulate traffic based on real-time conditions.
- Integrate security awareness into flow-control decisions.
- Optimize subscription behavior to reduce routing overhead.
- Leverage intelligent algorithms for predictive and autonomous management.

8.6. Final Remarks

In conclusion, while MQTT remains a foundational protocol for IoT communication, its effective deployment in modern, large-scale environments requires a fundamental shift in design philosophy. The challenges of **security, scalability, and performance cannot be addressed independently**; instead, they must be treated as interconnected components of a unified system.

This review highlights that achieving secure, scalable, and resilient MQTT-based IoT systems depends on the development of **integrated, adaptive, and context-aware frameworks**. Addressing this need will not only improve system performance but also enhance the reliability and security of next-generation IoT infrastructures.

9. Acknowledgments

The authors gratefully acknowledge the University of Idaho Open Access Publishing Fund for its generous financial support that enabled publication of this article.

Special thanks are also due to Professor Terry Saul for his sustained academic direction, thoughtful feedback, and constructive insights during the research process.

The authors further express gratitude to Professor Yong Wang, Chair of the Computer Science Department at the University of Idaho, for his guidance, professional advice, and institutional support.

Authors

Nael Radwan is a Ph.D. Candidate, in the Department of Computer Science at the University of Idaho, USA. He received his Bachelor of Science in Computer Science from Philadelphia University (1999), a Higher Diploma in Computer Science from Amman Arab University (2002), and a Master's degree in Computer Science from Amman Arab University (2005).

He began his academic career in 2000 at Al-Balqa Applied University in Jordan as an instructor in the Department of Information Technology. In 2005, he joined Al-Quds Open University and later the Arab Open University in Jeddah, Saudi Arabia, as a lecturer in the Information Systems Department. In 2010, he joined King Abdulaziz University as a lecturer in the Department of Computer Science, where he served until 2021.

In 2021, he moved to the United States to pursue his Ph.D. at the University of Idaho, where he is currently working as a Research Assistant. His research interests include computer networks, Internet of Things (IoT) systems, cybersecurity, and secure communication protocols. His academic profile is available at: https://drnaelradwan.academia.edu/

 doi:10.20944/preprints202604.0222.v1

Dr. Sheldon is currently a professor of computer science (CS) at the U. Idaho, has 35+ years of experience from academia, non-profit and industry sectors in various roles working on a diverse set of CS, data analytics/AI and cyberspace problems: Challenges related to AI and information security, resilience, deception, cybercrimes exploiting AI, trust and explain ability, OpSec, embedded avionics/vehicular software safety and security (e.g., CPS resiliency, intelligent transportation, IoT/Smart city/Cloud, energy delivery systems, supply chain, and cryptographic key management). His research projects as of late, are sourced from the convergence of software engineering, information assurance, leveraging data science and analytics (ML/DL). He received the Sigma Xi research and UT-Battelle key contributor, R&D 100 and significant event awards and is a life Sr. member of IEEE and ACM (+ASEE member). He has two BS degrees from the UMN Twin Cities and MS/PhD in CS from U. Texas at Arlington.

References

1. L. Da Xu, W. He, and S. Li, "Internet of things in industries: A survey," *IEEE Trans. Ind. Inform.*, vol. 10, no. 4, pp. 2233–2243, 2014.
2. A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Commun. Surv. Tutor.*, vol. 17, no. 4, pp. 2347–2376, 2015.
3. A. Banks and R. Gupta, "MQTT Version 3.1. 1," *OASIS Stand.*, vol. 29, p. 89, 2014.
4. N. M. Radwan, "A study: The future of the internet of things and its home applications," *Int. J. Comput. Sci. Inf. Secur. IJCSIS*, vol. 18, no. 1, 2020, Accessed: Apr. 01, 2026. [Online]. Available: https://www.researchgate.net/profile/Nael-Radwan/publication/340451499_A_Study_The_Future_of_the_Internet_of_Things_and_its_Home_Applications/links/6025e91592851c4ed5668a52/A-Study-The-Future-of-the-Internet-of-Things-and-its-Home-Applications.pdf
5. O. Standard, "MQTT Version 5.0," *Retrieved June*, vol. 22, no. 2020, p. 1435, 2019.
6. S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in Internet of Things: The road ahead," *Comput. Netw.*, vol. 76, pp. 146–164, 2015.
7. M. Ammar, G. Russello, and B. Crispo, "Internet of Things: A survey on the security of IoT frameworks," *J. Inf. Secur. Appl.*, vol. 38, pp. 8–27, 2018.
8. J. Qiu, Z. Tian, C. Du, Q. Zuo, S. Su, and B. Fang, "A survey on access control in the age of internet of things," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 4682–4696, 2020.
9. W. Eddy, "Rfc 9293: Transmission control protocol (tcp)." RFC Editor, 2022. Accessed: Apr. 01, 2026. [Online]. Available: https://ftp.st.ryukoku.ac.jp/pub/internet/rfc/rfc9293.pdf
10. A. Lundgren and M. Landin, "Data Flow Control in IoT: A Study on Throttling Implementations in MQTT." 2025. Accessed: Apr. 01, 2026. [Online]. Available: https://www.diva-portal.org/smash/record.jsf?pid=diva2:1994519
11. B. Mishra and A. Kertesz, "The use of MQTT in M2M and IoT systems: A survey," *Ieee Access*, vol. 8, pp. 201071–201086, 2020.
12. A. J. Hintaw, S. Manickam, M. F. Aboalmaaly, and S. Karuppayah, "MQTT Vulnerabilities, Attack Vectors and Solutions in the Internet of Things (IoT)," *IETE J. Res.*, vol. 69, no. 6, pp. 3368–3397, Aug. 2023, doi: 10.1080/03772063.2021.1912651.
13. W. Alsabbagh et al., "MQTT Protocol in Industrial Internet of Things: Today Challenges and Tomorrow Solutions," *Peter Langendoerfers Lab*, vol. 14, no. 8, pp. 1–31, 2021.
14. N. M. Radwan and J. Alves-Foss, "MQTT in Focus: Understanding the Protocol and Its Recent Advancements," *Int. J. Comput. Sci. Secur. IJCSS*, vol. 18, no. 1, pp. 1–14, 2024.
15. T. Begović, V. Čabarkapa, M. Ivković, and B. Popović, "MQTT Protocol in Smart Home Environments: Principles of Operation and Application," *Int. J. Electr. Eng. Comput.*, vol. 8, no. 2, pp. 53–61, 2024.

 doi:10.20944/preprints202604.0222.v1

16. F. Palmese, A. E. Redondi, and M. Cesana, "Adaptive quality of service control for mqtt-sn," *Sensors*, vol. 22, no. 22, p. 8852, 2022.
17. Y. Liu, D. Lan, Z. Pang, M. Karlsson, and S. Gong, "Performance evaluation of containerization in edge-cloud computing stacks for industrial applications: A client perspective," *IEEE Open J. Ind. Electron. Soc.*, vol. 2, pp. 153–168, 2021.
18. R. Qurishi and Z. Zhang, "Security Analysis of Popular MQTT Broker Platforms," 2025, Accessed: Apr. 01, 2026. [Online]. Available: https://odr.chalmers.se/bitstreams/f4aad2b4-618a-45cc-99d3-3e0f612e9bfb/download
19. N. Laaksonen, "MQTT suitability for performance-critical resource control communication," 2025, Accessed: Apr. 01, 2026. [Online]. Available: https://www.theseus.fi/handle/10024/905954
20. B. Mishra and A. Kertesz, "The use of MQTT in M2M and IoT systems: A survey," *Ieee Access*, vol. 8, pp. 201071–201086, 2020.
21. M. Algarni, M. Alkhelaiwi, and A. Karrar, "Internet of things security: A review of enabled application challenges and solutions," *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 3, 2021, Accessed: Apr. 01, 2026. [Online]. Available: https://www.researchgate.net/profile/Abdelrahman-Karrar-4/publication/350551718_Internet_of_Things_Security_A_Review_of_Enabled_Application_Challenges_and_Solutions/links/60b09bab299bf13438f009be/Internet-of-Things-Security-A-Review-of-Enabled-Application-Challenges-and-Solutions.pdf
22. M. Adam, M. Hammoudeh, R. Alrawashdeh, and B. Alsulaimy, "A survey on security, privacy, trust, and architectural challenges in IoT systems," *IEEE Access*, vol. 12, pp. 57128–57149, 2024.
23. S. Tian and V. G. Vassilakis, "On the efficiency of a lightweight authentication and privacy preservation scheme for MQTT," *Electronics*, vol. 12, no. 14, p. 3085, 2023.
24. M. K. M. AL-shammari, S. A. Abead, and H. H. Mahmoud, "Efficiency-Oriented Mutual Authentication Protocol for MCC Using Elliptic Curve Cryptography," *J. Internet Serv. Inf. Secur. JISIS*, vol. 16, no. 1, pp. 595–611, 2026.
25. Y. Yang, Y. Chen, F. Chen, and J. Chen, "Identity-based cloud storage auditing for data sharing with access control of sensitive information," *IEEE Internet Things J.*, vol. 9, no. 13, pp. 10434–10445, 2021.
26. A. Lundgren and M. Landin, "Data Flow Control in IoT: A Study on Throttling Implementations in MQTT." 2025. Accessed: Apr. 01, 2026. [Online]. Available: https://www.diva-portal.org/smash/record.jsf?pid=diva2:1994519
27. J. A. Garcia-Macias, "Transport in the IP-based Internet of Things: status report," *Procedia Comput. Sci.*, vol. 224, pp. 18–25, 2023.
28. Y. Liu, D. Lan, Z. Pang, M. Karlsson, and S. Gong, "Performance evaluation of containerization in edge-cloud computing stacks for industrial applications: A client perspective," *IEEE Open J. Ind. Electron. Soc.*, vol. 2, pp. 153–168, 2021.
29. R. Banno and Y. Watanabe, "Wildcard Topic Management Using Bloom Filter in Distributed MQTT Brokers," in *2025 IEEE 22nd Consumer Communications & Networking Conference (CCNC)*, IEEE, 2025, pp. 1–6. Accessed: Apr. 01, 2026. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10975906/
30. M. M. Vardhan, R. Tejdeep, P. Mishra, and V. A. Narayanan, "Secure and efficient communication for MQTT-based IoT systems," in *IoT Security*, Elsevier, 2026, pp. 203–217. Accessed: Apr. 01, 2026. [Online]. Available: https://www.sciencedirect.com/science/chapter/edited-volume/pii/B9780443341250000088
31. Y. Liu, "Investigating the Reliability and Security of the MQTT Protocol," PhD Thesis, 2022. Accessed: Apr. 01, 2026. [Online]. Available: https://digital.lib.washington.edu/researchworks/items/323c1d3f-8b70-4c76-be1c-720d3d46a20d
32. S. U. A. Laghari, W. Li, S. Manickam, P. Nanda, A. K. Al-Ani, and S. Karuppayah, "Securing MQTT ecosystem: Exploring vulnerabilities, mitigations, and future trajectories," *IEEE Access*, vol. 12, pp. 139273–139289, 2024.
33. S. U. A. Laghari, W. Li, S. Manickam, P. Nanda, A. K. Al-Ani, and S. Karuppayah, "Securing MQTT ecosystem: Exploring vulnerabilities, mitigations, and future trajectories," *IEEE Access*, vol. 12, pp. 139273–139289, 2024.

 doi:10.20944/preprints202604.0222.v1

34. S. U. A. Laghari, W. Li, S. Manickam, P. Nanda, A. K. Al-Ani, and S. Karuppayah, "Securing MQTT ecosystem: Exploring vulnerabilities, mitigations, and future trajectories," *IEEE Access*, vol. 12, pp. 139273–139289, 2024.
35. M. K. M. AL-shammari, S. A. Abead, and H. H. Mahmoud, "Efficiency-Oriented Mutual Authentication Protocol for MCC Using Elliptic Curve Cryptography," *J. Internet Serv. Inf. Secur. JISIS*, vol. 16, no. 1, pp. 595–611, 2026.
36. S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in Internet of Things: The road ahead," *Comput. Netw.*, vol. 76, pp. 146–164, 2015.
37. Y.-S. Yang, S.-H. Lee, W.-C. Chen, C.-S. Yang, Y.-M. Huang, and T.-W. Hou, "TTAS: Trusted token authentication service of securing SCADA network in energy management system for industrial Internet of Things," *Sensors*, vol. 21, no. 8, p. 2685, 2021.
38. Y. Liu, D. Lan, Z. Pang, M. Karlsson, and S. Gong, "Performance evaluation of containerization in edge-cloud computing stacks for industrial applications: A client perspective," *IEEE Open J. Ind. Electron. Soc.*, vol. 2, pp. 153–168, 2021.
39. J. Wu et al., "A Survey on Cloud-Edge-Terminal Collaborative Intelligence in AIoT Networks," Aug. 26, 2025, *arXiv*: arXiv:2508.18803. doi: 10.48550/arXiv.2508.18803.
40. B. Mishra and A. Kertesz, "The use of MQTT in M2M and IoT systems: A survey," *Ieee Access*, vol. 8, pp. 201071–201086, 2020.
41. A. Javed, M. Awais, M. Shoaib, K. S. Khurshid, and M. Othman, "Machine learning and deep learning approaches in IoT," *PeerJ Comput. Sci.*, vol. 9, p. e1204, 2023.
42. Y. Liu, J. Wang, J. Li, S. Niu, and H. Song, "Machine learning for the detection and identification of Internet of Things devices: A survey," *IEEE Internet Things J.*, vol. 9, no. 1, pp. 298–320, 2021.
43. A. Hassan, J. Aslam, S. Tahir, and I. Rashid, "Securing the Internet of Things: Exploring MQTT Vulnerabilities and Threats in Pakistan's IoT Landscape," in *2024 International Conference on Engineering & Computing Technologies (ICECT)*, IEEE, 2024, pp. 1–6. Accessed: Apr. 01, 2026. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10581385/