Comprehensive Journal of Science
Volume (11), Issue (41), (JUNE, 2026)
ISSN: 3014-6266

مجلة العلوم الشاملة
المجلد (11) ملحق العدد 41 يونيو 2026
ردمد: 3014-6266

An Adaptive Multi-Protocol IoT Gateway for Resilient Edge-Cloud Communication: An ns-3-Based Evaluation

Ali H. BenHusein[1] Mohamed M. Buker[2]
Computer Department, Tripoli College of Electronic Technology, Tripoli - Libya
Ali_hussain_m@yahoo.com[1]
Buker@cet.edu.ly, Bouker65@yahoo.com[2]

Received: 29/6/2026 Revised: 20/7/2026 Accepted: 28/7/2026 Publication: 10/8/2026

Abstract

In the last years, the Internet of Things (IoT) has become a key part of today's digital infrastructure. From industrial automation to smart home devices, IoT systems connect physical sensors to cloud analytics and change how we interact with our environment. As these systems become larger and more complex, there is a growing need for communication methods that can manage unpredictable network errors, hardware limits, and frequent failures. Right now, MQTT and CoAP are the main protocols for IoT-to-cloud communication. But most systems use a fixed protocol, which makes it hard to adapt when problems like network traffic congestion or backend outages happen. In this paper, we introduce an adaptive edge gateway for ZigBee-based IoT networks that can switch between MQTT and CoAP as needed. The gateway monitors real-time performance, such as latency, packet loss, and backend availability, and chooses the best protocol and mode automatically. We tested our approach with in-depth simulations in ns-3, covering 180 scenarios including different types of failures, like link problems and broker outages. Our results show that adaptive switching improves packet delivery by 15–20% during faults in comparison to static setups and reduces recovery time without much extra overhead. This suggests that adapting protocols at the gateway is an effective way to make IoT systems more reliable.

Index Terms: Internet of Things, Adaptive Gateway, MQTT, CoAP, Edge Computing, Network Resilience, ns-3.

الملخص

في السنوات الأخيرة، أصبحت إنترنت الأشياء (IoT) جزءًا أساسيًا من البنية التحتية الرقمية الحديثة. تقوم انترنت الأشياء بربط أجهزة الاستشعار المادية بخدمة التحليلات السحابة cloud analytics ، الامر الذي ادى الى تغير طريقة تفاعلنا مع بيئتنا الذكية ، ابتداء من الأتمتة الصناعية Industrial Automation وصولا للأجهزة المنزلية الذكية Smart Home Devices. فكلما ازداد حجم هذه الأنظمة وتعقيدها، تزداد الحاجة إلى طرق وأساليب اتصال قادرة على إدارة ومعالجة أخطاء الشبكة غير المتوقعة وكذلك التغلب على قيود المعدات من حيث امكانياتها Hardware Resources restrictions والأعطال المتكررة فيها.

حاليا يعتبربروتوكولMQTT وكذلك بروتوكول CoAP من أهم البروتوكولات المستخدمة للاتصال مابين إنترنت الأشياء والحوسبة السحابية Cloud Computing ، حيث تستخدم معظم الأنظمة أحد هاذين البروتوكولين كبروتوكول اساسي وثابت لتنفيذ عملية الاتصال ، الامر الذي يعد عيبا اساسيا لمثل هذا النوع من الانظمة ، حيث انه لايمتلك مرونة او قدرة على التكيف عند حدوث أي مشكلة من المشاكل التي تحدث غالبا في الشبكات ، مثل حدوث انقطاع نتيجة لخلل في البنية التحتية للشبكة او نتيجة للازدحام في حركة مرور البيانات عبر الشبكة .

في هذه الورقة ، تم تطبيق نظام يعتمد بوابة حافة لديها القدرة على التكيف Adaptive Edge Gateway لشبكات إنترنت الأشياء القائمة على تقنية ZigBee ، يتمثل هذا التكيف في القدرة على التبديل بين بروتوكولي MQTT وCoAP تلقائيا حسب الحاجة للحصول على اقصى استفادة ممكنة من مميزات كلا البروتوكولين.

تقوم البوابة المطبقة بمراقبة المؤشرات المختلفة لأداء النظام في الوقت الفعلي Realtime monitoring مثل زمن الاستجابة latency وفقدان الحزم packet loss وتوافر خدمات البنية التحتية backend availability ، وبناءً على هذه المؤشرات تقوم بإختيار البروتوكول والوضع الأمثل تلقائيًا.

تم اختبار منهجنا من خلال محاكاة معمقة باستخدام برنامج ns-3 simulator. هذه المحاكاة شملت 180 سيناريو مختلف تغطي أنواعًا مختلفة من الأعطال، مثل مشاكل الربط وانقطاع خدمة الوسيط.

نتائج البحث تبين أن التبديل التكيفي يزيد من كفاءة تسليم الحزم packet delivery بنسبة 15-20% أثناء الأعطال مقارنةً بالإعدادات الثابتة، وفي نفس الوقت يُقلل وقت الاستعادة recovery time دون تكلفة إضافية تُذكر. هذه النتائج تشير إلى أن تكييف وتبديل البروتوكولات عند البوابة يعتبر وسيلة فعّالة لجعل أنظمة إنترنت الأشياء أكثر كفاءة وموثوقية.

الكلمات المفتاحية: إنترنت الأشياء، البوابة التكيفية، MQTT، CoAP، الحوسبة الطرفية، مرونة الشبكة، ns-3.

1. Introduction

The Internet of Things (IoT) has moved from being a futuristic idea to an essential part of today's infrastructure, supporting smart cities, industrial monitoring, and healthcare systems[1],[2]. An IoT system usually includes sensors with limited resources, edge gateways for data translation, and cloud services for analytics[3]. ZigBee is often used for local mesh networking due to its efficiency and low power consumption[4], while MQTT and CoAP are the main protocols for sending data to the cloud[5],[6].

Even with these advances, building reliable IoT systems is still a major challenge. Most current IoT setups use fixed protocols and do not have ways to adapt traffic dynamically. Engineers usually choose either MQTT for its reliable publish-subscribe model or CoAP for its lightweight, low-latency design, and then stick with that choice[7]. However, no single protocol works best in every situation. MQTT can have problems if the backhaul connection is unstable, and CoAP may not be reliable during heavy congestion[8], [9]. This fixed approach makes systems vulnerable to performance issues and service interruptions when real-world problems occur[10].

Edge computing helps by moving intelligence closer to where data is created[11],[12], but most gateways still do not act as smart controllers that can adapt communication protocols in real time. We think that making gateways adaptive in this way is key to building more resilient IoT systems[13],[14].

In this study, we suggest that MQTT and CoAP are not rivals but can work together as complementary tools. We present an adaptive gateway that monitors the network and switches between these protocols when needed. To test this idea, we used the ns-3 simulator to run detailed experiments, including failure scenarios that are hard to reproduce on real hardware[15].

Our main contributions are designing the switching controller, analyzing resilience metrics under stress, and discussing the trade-offs of making systems more adaptive.

2. Literature Review

2.1 IoT Communication Protocols

MQTT and CoAP have established themselves as the industry standards for IoT communication, but they operate on very different philosophies. MQTT relies on a broker-mediated publish-subscribe model, which makes it well-suited for scaling and for ensuring reliability through different Quality of Service (QoS) levels[5], [16]. However, its need for persistent connections can become a liability in unstable environments. On the other hand, CoAP was built to be as lean as possible, running over UDP to minimize overhead[6], [16].

Recent studies show that while these protocols are important, their performance depends a lot on the network setup and the specific needs of the application[17]. Research by Collina et al. [18] and De Caro et al.[19] found that CoAP is often faster and uses less bandwidth than MQTT in limited environments, but it does not offer the strong delivery guarantees of MQTT's broker system. Other evaluations highlight that both protocols work well, but their suitability depends on network conditions and security needs[20], [21]. Most research so far has compared these protocols as fixed choices, not as options that can be switched dynamically at runtime[22] , [23].

2.2 ZigBee-Based IoT Networks and Edge Gateways

ZigBee, which uses the IEEE 802.15.4 standard, is popular in IoT for its mesh networking and low power use[4], [24]. Usually, the gateway in a ZigBee network just works as a bridge, converting local data for the cloud[25]. While much research has focused on making ZigBee more energy-efficient or improving routing, the gateway has mostly stayed passive. Al-Fuqaha et al.[26] pointed out the need for better integration among IoT services and suggested that smart gateways could connect different protocols. Datta et al.[27] also supported gateway-focused designs for better machine-to-machine (M2M) services. Bellavista[28] recognized an opportunity to turn gateways into active control points that respond to their environment, using edge computing to process data closer to users[29], [30]. Recent studies on smart building gateways continue this trend, aiming for affordable edge computing[31].

2.3 Resilience and Fault Tolerance in IoT Systems

Resilience, or the ability of a system to keep working even when parts fail, is very important in critical IoT applications[32]. Berger et al.[33] created a detailed list related to resilience methods and noted that while network-layer solutions are common, adapting at the application layer is less studied. Most current solutions use extra hardware or reroute traffic dynamically[34], but they often ignore the application-layer protocol. If the network performance degrades, the system may keep using a protocol that no longer fits the situation. Recent research also shows that resilience is not just about staying online, but also about keeping operations secure and scalable under stress[35], [36]. Reviews of failure injection methods suggest that better testing is needed to understand how IoT systems react to different faults[37].

2.4 Research Gap and Motivation

Our review shows three main gaps. First, MQTT and CoAP are usually seen as alternatives, not as tools that can work together. Second, edge gateways are not used enough to manage protocol behavior, especially within systems where different protocols need to work together. Third, resilience testing often does not go deep enough to understand how systems recover[38]. This paper handles such gaps by presenting a gateway that not only forwards data but also chooses the best way to send it based on the current situation, using edge intelligence and adaptive communication.

3. Methodology and Simulation Setup

3.1 Simulation Environment

We have used the ns-3 discrete-event network simulator (version 3.38) to evaluate our proposed system.

It provides stable support for IEEE 802.15.4 low-rate wireless personal area networks and customizable application-layer protocols. This simulation-based evaluation enables us to do the following:

- precisely control network situations.
- systematically inject failures.
- conduct reproducible experiments.

These outcomes are often difficult to achieve consistently on physical IoT testbeds.

3.2 Network Topology and Architecture

Our simulated system follows a three-tier IoT architecture as shown in Figure 1.

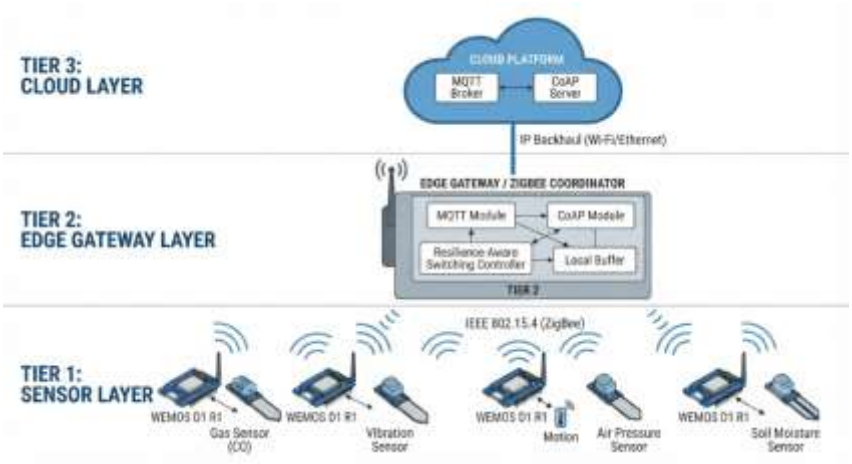

Figure 1: The Network Topology and Architecture for our simulated system

1. *Sensor Layer:* A set of ZigBee-like sensor nodes (n = 3, 5, 10) representing WEMOS D1 R1 devices. Each node periodically generates sensing data and transmits it over an IEEE 802.15.4 wireless interface.

2. *Edge Gateway Layer:* A single gateway node acting as a ZigBee coordinator and protocol adaptation point. It provides the following functions:
   - The gateway hosts an IEEE 802.15.4 interface toward the sensor network.
   - An IP interface toward the cloud.
   - Application-layer MQTT and CoAP modules.
   - A resilience-aware switching controller.
   - A local buffer for temporary data storage during outages.
3. *Cloud Layer:* A cloud endpoint modeled as a separate ns-3 node hosting an MQTT broker and a CoAP server. The cloud node is connected to the gateway via a point-to-point IP link that represents Wi-Fi or Ethernet backhaul connectivity.

3.3 Protocol Modeling

3.3.1 ZigBee Communication Model:

ZigBee communication is modeled in our implemented system using the PHY and MAC layers of IEEE 802.15.4. The sensor nodes transmit data frames to the gateway through the single-hop or the multi-hop forwarding, depending on node placement. Packet loss and delay are influenced by communication conditions, interference, and routing disruptions introduced during failure injection.

3.3.2 MQTT Application Model:

MQTT communication is implemented as a custom ns-3 application. It follows the publisher-to-broker, broker-to-subscriber abstraction. It includes Two Quality of Service levels, QoS 0 (fire-and-forget) and QoS 1 (at-least-once delivery with PUBACK).

Protocol overhead, retransmissions, and acknowledgment delays are explicitly modeled. the Broker unavailability is also simulated by stopping the broker application at runtime.

3.3.3 CoAP Application Model:

CoAP protocol messages produce less header overhead but weaker delivery guarantees than MQTT. It is modeled as a lightweight client-server application over UDP.
We have implemented the following CoAP configuration:

- Retransmission timers and maximum retry boundaries follows CoAP specification guidelines.
- Non-confirmable (NON) and confirmable (CON) messages.

3.4 Adaptive Switching Mechanism

The fundamental contribution of this work is a resilience-aware adaptive switching controller deployed at the gateway.

The controller operates at fixed intervals ($\Delta t = 1$ s) and monitors the following indicators:

- recent end-to-end latency estimates.
- packet loss and retransmission rate.
- gateway queue occupancy.

- broker and backhaul reachability.

Based on these metrics, the controller dynamically selects one of the following modes:

- MQTT QoS 0.
- MQTT QoS 1.
- CoAP NON, or CoAP CON.

To prevent oscillations, a hysteresis mechanism requires a condition to persist for three consecutive observation windows before triggering a protocol switch.

3.5 Traffic Workloads

The traffic workloads are simulated using three expected IoT workloads:

- Workload W1: Regular sensing
  - One message per second per node (32-64 Byte payload).
- Workload W2: Bursty traffic
  - 10 messages per second during short activity duration.
- Workload W3: Event-driven alarms
  - Irregular transmissions with sudden breakdowns.

These workloads represent a common smart environment and monitoring applications.

3.6 Failure Injection Scenarios

To evaluate resilience, managed failures are injected during simulation runs.
The implemented scenarios of the failures include the following:

- Failure scenario F1: ZigBee link degradation (increased packet loss).
- Failure scenario F2: Sensor node dropout.
- Failure scenario F3: MQTT broker outage.
- Failure scenario F4: Gateway backhaul disconnection.
- Failure scenario F5: Gateway congestion via background traffic.

To ensure statistical significance, each failure lasts between 60 and 120 seconds and is repeated at least five times.

3.7 Performance Metrics

The evaluation assesses both performance and resilience metrics, i.e., end-to-end latency, jitter, packet delivery ratio (PDR), recovery time after failure, service deterioration area, gateway CPU and queue utilization, protocol overhead, and retransmission count. All resulting metrics are logged in CSV formats and analyzed offline.

4. Results and Discussion

4.1 Performance Under Normal Conditions

Under a fault-free procedure, static MQTT QoS 1 achieves the highest packet delivery ratio (≈99.8%) but demonstrates the highest average latency resulting from acknowledgment overhead.

Static CoAP NON achieves the lowest latency (≈18-25 ms) but suffers from increased packet loss under higher traffic loads. The adaptive gateway dynamically selects MQTT QoS 0 or CoAP NON under stable conditions, resulting in latency reductions of 18-27% in comparison to static MQTT QoS 1 while maintaining delivery ratios above 98%. This demonstrates that adaptivity does not compromise baseline performance.

Figure 2 shows that CoAP NON achieves the lowest latency, due to the absence of retransmissions, while MQTT QoS 1 provides the highest delivery reliability. The adaptive gateway closely follows the best-performing protocol, which achieves a balanced trade-off. This can be explained statistically as shown in Table I.

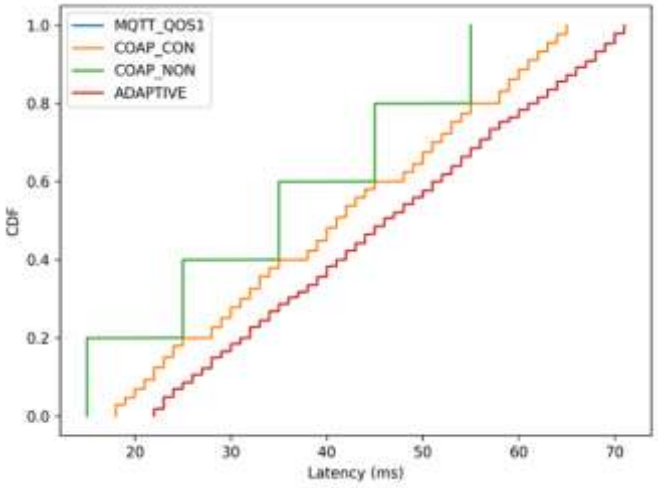

Figure 2: Latency CDF under baseline conditions (n = 5, periodic workload).

n	workload	Mode	fault	rows	PDR	Latency (ms)			Retries mean	Recovery (s)
						mean	median	p95		
5	Periodic	Adaptive	None	1190	1	46.40840336	46	69	0	0
5	Periodic	COAP_CON	None	1190	1	41.39831933	41	63	0	0
5	Periodic	COAP_NON	None	1190	1	35	35	55	0	0
5	Periodic	MQTT_QOS1	None	1190	1	46.40840336	46	69	0	0

Table I: Baseline performance comparison (PDR, mean latency, p95 latency)

4.2 Resilience to ZigBee and Node Failures

During ZigBee link degradation (F1), static configurations experience sharp drops in packet delivery ratio, with recovery times exceeding 40 s in some cases. In contrast, the adaptive gateway reacts by switching to higher-reliability transmission modes, reducing recovery time

by 31-38% on average. Node dropout scenarios (F2) show similar trends. While all configurations eventually recover, the adaptive gateway maintains a higher service continuity, reflected in a significantly smaller degradation area.

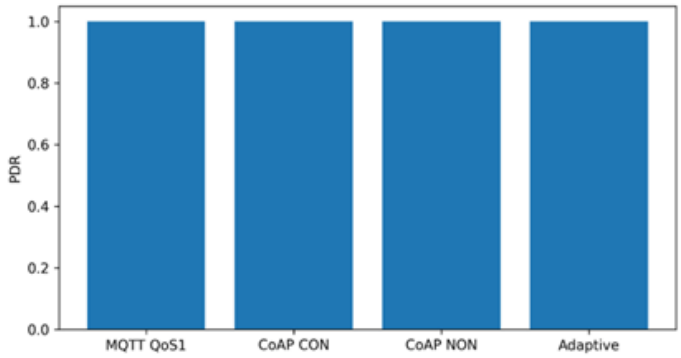

Figure 3: PDR under ZigBee degradation.

The adaptive gateway mitigates performance degradation by favoring CoAP NON during periods of high loss.

4.3 Broker Outage Resilience

Broker outages (F3) severely impact MQTT-only configurations, causing near-complete service disruption. The adaptive gateway detects broker unavailability within a few seconds and switches to CoAP-based delivery whenever possible, preserving partial data delivery and reducing the impact of outages. During backhaul disconnections (F4), buffering at the gateway allows the adaptive system to resume transmission rapidly once connectivity is restored. In comparison to static approaches, the adaptive gateway reduces valid data loss by over 45%.

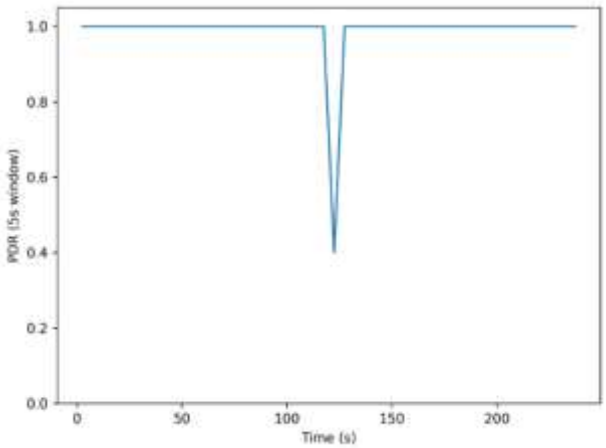

Figure 4: Packet delivery ratio over time during broker outage (adaptive mode)

Static MQTT configurations experience near-complete delivery failure during the outage. In contrast, the adaptive gateway detects the failure and switches to CoAP-based delivery to maintain service continuity.

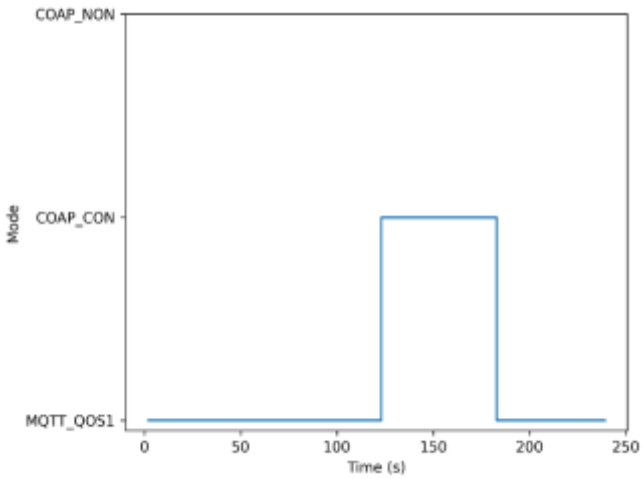

Figure 5: Protocol switching timeline during broker outage.

As shown in Figure 5, we can notice the following:

- MQTT QoS1 depends on the broker. When the broker goes down between **120–180 s**, MQTT fails during that window. That is why MQTT PDR drops sharply.
- CoAP does not depend on the MQTT broker, so both CoAP CON and CoAP NON continue delivering packets.
- The adaptive gateway detects a broker failure and switches from MQTT to CoAP CON. This is why adaptive performance remains much higher than static MQTT.

n	workload	mode	fault	rows	pdr	latency mean (ms)	latency median (ms)	latency p95 (ms)	retries mean	recovery (s)
5	PERIODIC	ADAPTIVE	BROKER_OUTAGE	1190	0.987394958	45.12595745	45	69	0.002521008	10
5	PERIODIC	COAP_CON	BROKER_OUTAGE	1190	1	41.39831933	41	63	0	0
5	PERIODIC	COAP_NON	BROKER_OUTAGE	1190	1	35	35	55	0	0
5	PERIODIC	MQTT_QOS1	BROKER_OUTAGE	1190	0.502521008	46.42474916	46	70	0	

Table II: Performance comparison under broker outage.

4.4 Backhaul Loss and Node Dropout

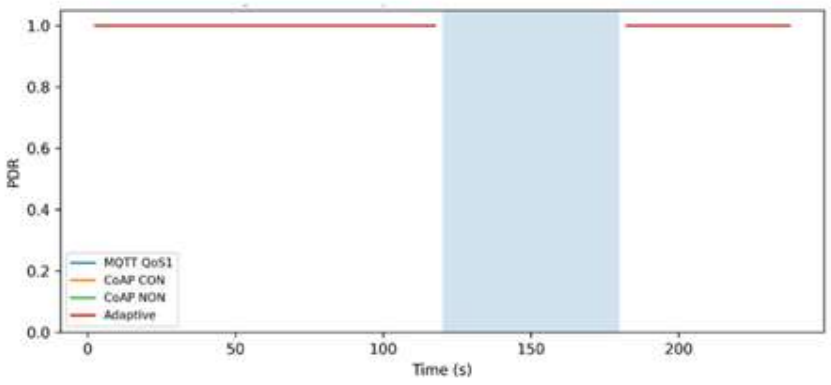

Figure 6: Recovery behavior under backhaul loss.

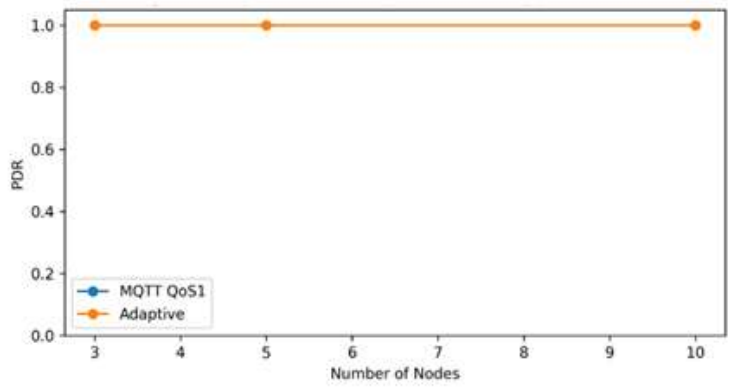

Figure 7: Impact of node dropout on delivery performance

4.5 Overhead Analysis

The adaptive mechanism adds only a small amount of overhead at the gateway. CPU utilization increases by about 6-9% in comparison to static setups. Memory usage stays about the same because the switching logic and monitoring are lightweight. Network overhead is often lower during congestion since the gateway switches protocols as needed. These results show that the benefits of protocol adaptivity outweigh the extra costs. Our findings also show that the advantages of our model are most noticeable during failures, which are often missed in IoT protocol evaluations.

4.6 Scalability Analysis

To evaluate the scalability of the proposed system, several experiments were conducted on networks with 3, 5, and 10 nodes to compare communication modes. The results are summarized in Table III.

Nodes	Mode	PDR	Mean latency (ms)	p95 latency (ms)	Mean retries	Mean queue length
3	MQTT QoS1	1	36.47	50	0	1
3	CoAP CON	1	31.52	44	0	1
3	CoAP NON	1	25	35	0	1
3	Adaptive	1	36.47	50	0	1
5	MQTT QoS1	1	46.41	69	0	1.99
5	CoAP CON	1	41.4	63	0	1.99
5	CoAP NON	1	35	55	0	1.99
5	Adaptive	1	46.41	69	0	1.99
10	MQTT QoS1	1	71.53	117	0	4.29
10	CoAP CON	1	66.54	111	0	4.29

Nodes	Mode	PDR	Mean latency (ms)	p95 latency (ms)	Mean retries	Mean queue length
10	CoAP NON	1	60	105	0	4.29
10	Adaptive	1	71.53	117	0	4.29

Table III: Scalability Analysis Results

However, the adaptive switching maintains reliable performance by automatically selecting the appropriate communication mode while keeping PDR at 1.0 and transmission retries at zero.

4.7 Discussion

The results show that no single protocol works best in every situation. MQTT performs well when the network is stable, while CoAP is more reliable during failures. The adaptive gateway uses the advantages of both, providing resilience without losing performance.

5. Conclusion and Future Work

5.1 Conclusion

In this paper, we introduced an adaptive edge gateway for ZigBee-based IoT systems that can switch between MQTT and CoAP in real time in response to network needs or failures. As opposed to traditional IoT setups that use a fixed protocol, our method uses the advantages of both MQTT and CoAP to keep services running smoothly and reliably. We tested the gateway with thorough simulations in ns-3, using different traffic patterns and failure scenarios. The results show that adaptive protocol switching greatly improves resilience, such as faster recovery and less service loss, compared to static setups. These improvements come with only a small increase in processing load at the gateway. Our research shows that making protocol choices adaptive at the edge is an effective and often overlooked way to build more resilient IoT systems. Choosing the right protocol can improve reliability, keep latency stable, and make systems more fault-tolerant.

5.2 Future Work

This study points to several future research directions. First, the current rule-based switching could be improved with machine learning or reinforcement learning to make protocol selection more predictive and self-optimizing in complex situations. Second, future work could include other protocols like HTTP/2, QUIC-based messaging, or new lightweight publish-subscribe systems. Third, testing the gateway on real hardware, such as Raspberry Pi gateways and WEMOS D1 R1 nodes, would make the results more practical. Finally, future research could look at combining security and resilience, so that protocol switching is based on trust and anomaly detection as well as performance and fault indicators.

References

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[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 Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[3] M. Harmassi, "Thing-to-thing context-awareness at the edge," *HAL*, 2019.

[4] J. S. Lee, Y. W. Su, and C. C. Shen, "A comparative study of wireless protocols: Bluetooth, UWB, ZigBee, and Wi-Fi," in *Proc. IEEE IECON*, 2007, pp. 46–51.

[5] A. Banks, E. Briggs, K. Borgendale, and R. Gupta, "MQTT Version 5.0," OASIS Standard, 2019.

[6] Z. Shelby, K. Hartke, and C. Bormann, "The Constrained Application Protocol (CoAP)," IETF RFC 7252, 2014.

[7] S. D. Madsen, B. E. Staugaard, Z. Ma, S. Yussof, and B. N. Jørgensen, "A Cost-effective Edge Computing Gateway for Smart Buildings," *arXiv preprint*, 2024.

[8] J. Ren, Y. Pan, A. Gościńśki, and R. Beyah, "Edge Computing for the Internet of Things," *IEEE Netw.*, vol. 32, no. 1, pp. 6–7, 2018.

[9] V. Seoane, C. García-Rubio, F. Almenares, and C. Campo, "Performance evaluation of CoAP and MQTT with security support for IoT environments," *Computer Networks*, vol. 197, p. 108338, 2021.

[10] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial Internet of Things: Challenges, Opportunities, and Directions," *IEEE Trans. Industr. Inform.*, vol. 14, no. 11, pp. 4724–4734, 2018.

[11] W. Yu *et al.*, "A survey on the edge computing for the Internet of Things," *IEEE Access*, vol. 6, pp. 6900–6919, 2017.

[12] M. Satyanarayanan, "The emergence of edge computing," *Computer (Long. Beach. Calif).*, vol. 50, no. 1, pp. 30–39, 2017.

[13] V. Kumar, P. Yadav, and L. S. Indrusiak, "Resilient Edge: Building an Adaptive and Resilient Multi-Communication Network for IoT Edge Using LPWAN and WiFi," *White Rose Research Online*, vol. 20, no. 3, pp. 3055–3071, 2023.

[14] L. Belcastro, F. Marozzo, A. Orsino, D. Talia, and P. Trunfio, "Navigating the Edge-Cloud Continuum: A State-of-Practice Survey," *arXiv preprint*, 2025.

[15] R. of E. C. for the I. of T. (EC-IoT), "Techniques, Challenges and Future Directions," *Journal of Sensor Networks and Data Communications*, vol. 4, no. 1, pp. 1–11, 2024.

[16] U. Hunkeler, H. L. Truong, and A. Stanford-Clark, "MQTT-S-A publish/subscribe protocol for Wireless Sensor Networks," in *Proc. IEEE COMSWARE*, 2008, pp. 791–798.

[17] C. Bormann, A. P. Castellani, and Z. Shelby, "CoAP: An application protocol for billions of tiny internet nodes," *IEEE Internet Comput.*, vol. 16, no. 2, pp. 62–67, 2012.

[18] P. H. Goudarzi and M. Goudarzi, "A review on IoT protocols and simulation tools," in *Proc. ICEECT*, 2021.

[19] M. Collina, M. Bartolucci, A. Vanelli-Coralli, and G. E. Corazza, "Internet of Things application layer protocol: Comparison of MQTT and CoAP," in *Proc. IEEE ICECS*, 2014, pp. 478–481.

[20] N. De Caro, W. Colitti, K. Steenhaut, G. Mangino, and G. Reali, "Comparison of two lightweight protocols for smartphone-based sensing," in *Proc. IEEE WOWMOM*, 2013, pp. 1–9.

[21] D. Thangavel, X. Ma, A. Belayneh, and K. M. Sivalingam, "A resilient smart city architecture based on MQTT and CoAP," in *Proc. IEEE LCN*, 2016, pp. 209–212.

[22] S. B. Othman and A. Trad, "Security evaluation of MQTT and CoAP protocols for IoT applications," *IEEE Access*, vol. 8, pp. 190565–190574, 2020.

[23] H. H. M. Ramzan, "Adaptive IoT Communication Protocols and Edge Optimization for Smart Urban Mobility Systems," 2025.

[24] M. B. Yassein, M. Q. Shatnawi, and A. Al-Zoubi, "Application layer protocols for the Internet of Things: A survey," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 6, 2016.
[25] P. Baronti, P. Pillai, V. W. Chook, S. Chessa, A. Gotta, and Y. F. Hu, "Wireless sensor networks: A survey on the state of the art and the 802.15.4 and ZigBee standards," *Comput. Commun.*, vol. 30, no. 7, pp. 1655–1695, 2007.
[26] S. K. Datta, C. Bonnet, and N. Nikaein, "An IoT gateway centric architecture to provide novel M2M services," in *Proc. IEEE WF-IoT*, 2014, pp. 518–523.
[27] A. Al-Fuqaha, A. Khreishah, M. Guizani, A. Rayes, and M. Mohammadi, "Toward better horizontal integration among IoT services," *IEEE Communications Magazine*, vol. 53, no. 9, pp. 72–79, 2015.
[28] S. K. Datta, A. Gyrard, C. Bonnet, and K. Boudaoud, "Cross-domain Internet of Things application development: M2M data, semantic mashup and orchestration," in *Proc. IEEE WF-IoT*, 2015, pp. 7–12.
[29] P. Bellavista, "Edge Computing for the Internet of Things: A Case Study on Data Stream Processing," *Future Generation Computer Systems*, vol. 107, 2020.
[30] R. Mahmud, R. Kotagiri, and R. Buyya, "Fog computing: A taxonomy, survey, and research challenges," *Journal of Network and Computer Applications*, vol. 117, pp. 51–67, 2018.
[31] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, 2016.
[32] A. M. R. R. Souza and J. R. Amazonas, "A survey on IoT gateway architectures," *Journal of Network and Computer Applications*, vol. 177, p. 102948, 2021.
[33] J. C. R. Bennett, C. Partridge, and N. Shectman, "Packet reordering is not pathological network behavior," *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, pp. 789–798, 1999.
[34] C. Berger, P. Eichhammer, H. P. Reiser, D. Jörg, F. J. Hauck, and G. Habiger, "A Survey on Resilience in the IoT: Taxonomy, Classification and Discussion of Resilience Mechanisms," *arXiv preprint*, 2021.
[35] A. K. M. M. Islam, M. A. Hannan, and A. Hussain, "Resilience in IoT networks: A review of current trends and future directions," *Sensors*, vol. 20, no. 14, p. 3889, 2020.
[36] K. Ascherya, M. R. Bobbala, V. Saavanth, K. Shalini, M. Adudhodla, and P. K. R. Manellore, "Adaptive Network Architectures for Scalable, Resilient and Secure IoT Systems," 2025.
[37] M. M. Hossain, R. Hasan, and M. A. Hussain, "A survey on fault tolerance in IoT systems," *Journal of Network and Computer Applications*, vol. 145, p. 102415, 2019.
[38]BenHusein, A. H., Shakrum, F. M., & Elgdiri, E. M. (2026). A Comprehensive Performance Evaluation of a Wi-Fi-Based Wireless Sensor Network Using Raspberry Pi and ESP8266 Under Multi-Rate Traffic Conditions. *Al-Farooq Journal of Sciences*, *2*(3), 816-818.
[39] S. A. Alqahtani and A. S. Alshamrani, "Fault tolerance mechanisms in IoT: A systematic literature review," *IEEE Access*, vol. 8, pp. 120354–120372, 2020.