IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Journal Européen des Systèmes Automatisés
Vol. 59, No. 7, July, 2026, pp. 1933-1944
Journal homepage: http://iieta.org/journals/jesa

Performance Benchmarking of IoT Edge Computing Versus Cloud Computing: A Controlled Simulation-Based Study Using Node-RED and the Intel Berkeley Sensor Dataset

Safa Altaie[1], Sana Sami Mohammed Tahir[1], Mahmood Alfathe[2*]

[1] Computer Center, University of Mosul, Mosul 41002, Iraq
[2] College of Information Technology, Nineveh University, Mosul 41001, Iraq

Corresponding Author Email: Mahmood.alfathe@uoninevah.edu.iq

https://doi.org/10.18280/jesa.590710

Received: 5 April 2026
Revised: 6 June 2026
Accepted: 14 June 2026
Available online: 31 July 2026

Keywords:
Internet of Things, edge computing, cloud computing, Message Queuing Telemetry Transport, Node-RED, latency, jitter, packet loss

ABSTRACT

The rapid proliferation of Internet of Things (IoT) devices imposes stringent Quality-of-Service (QoS) requirements on underlying computing infrastructure — particularly with respect to latency, jitter, throughput, and packet loss. Existing comparative studies of edge and cloud computing architectures are predominantly constrained by hardware-specific testbed confounders, selective single-metric reporting, and the absence of statistical replication, collectively undermining the reliability and reproducibility of published performance claims. This research corrects previous flaws via controlled, multi-metric, statistically verified trials programmed inside Node-RED alongside Eclipse Mosquitto serving as Message Queuing Telemetry Transport (MQTT) server software systems. Projects integrate Intel Berkeley Laboratory Wireless Sensor Network collection items representing accurate IoT communication message payloads. Latency spans mimic empirical lognormal metric values (Local Area Network (LAN): $\mu = 3.58$, $\sigma = 0.45$; Wide Area Network (WAN): $\mu = 4.61$, $\sigma = 0.55$) derived from official measurement studies, while drop rates observe specific ITU-T G.1010 standard benchmarks provided within (LAN: 0.1%; WAN: 1.0%). Thirty independent runs with fixed random seeds are executed per architecture per metric, enabling 95% Confidence Interval (CI) construction and Welch's t-test significance testing. Edge nodes return mean latency of 39.69 ms (±0.024 ms) whereas cloud servers achieved 116.87 ms (±0.092 ms) while representing (IF = 2.94×; $p < 0.001$), including P95 latency of 75.17 ms over 248.19 ms, jitter readings equaling 19.80 ms versus 70.72 ms (IF = 3.57×; $p < 0.001$), plus network packet loss sitting at 0.101% against 1.002% (IF = 9.90×; $p < 0.001$). Throughput values (1.561 versus 1.547 KB/s; IF = 1.01×) stay statistically significant ($p < 0.001$) yet architecturally minor, confirming how packet loss instead of raw bandwidth governs real-time overall final effective network throughput differential throughout controlled tests. All differences are statistically significant at $p < 0.001$. A hybrid three-tier edge-cloud architecture is proposed and validated as the optimal deployment pattern for latency-sensitive IoT applications.

1. INTRODUCTION

Internet-connected hardware currently defines our modern digital ecosystem. Recent Cisco research predicts networked devices will generate massive amounts totaling 79.4 zettabytes of total global traffic per year by 2025 [1]. With the rapid rise of the Internet of Things (IoT), there is an urgent need for scalable and diverse computing platforms able to process this data stream with the required quality of service. Previously, cloud computing was the default solution for storing and processing data retrieved from IoT devices because of its almost unlimited scalability, centralized management, and affordable storage [2]. However, it is not suitable for processing data with low latency because it needs to send information via the Wide Area Network (WAN) to distant servers. There is a growing interest in deploying computations closer to the source of data to reduce WAN delays and latencies. Nevertheless, routing IoT-generated data through WANs to geographically remote data centers introduces non-negligible latency and jitter penalties, rendering cloud-centric architectures unsuitable for mission-critical applications such as industrial automation, autonomous vehicles, surgical robotics, and real-time environmental monitoring [3, 4]. Edge nodes can host applications and data to ensure reliable communication and provide low-latency responses with latency below 100 ms [5]. The benefits of using edge computing were confirmed by analytical modeling, which showed that it is possible to reduce latency by moving some of the processing tasks closer to the data source while minimizing the consumption of power [5]. Edge computing can fulfill this need by bringing cloud infrastructure closer to IoT devices. The edge computing architecture standard, proposed by the European Telecommunications Standards Institute (ETSI), defines edge computing as a paradigm that aims to meet the demand for distributed cloud computing at the network edge [6].

Despite a growing body of literature, three critical methodological deficiencies persist across existing edge-cloud benchmarking studies. First, methodological heterogeneity — studies conducted on divergent hardware configurations and network topologies — makes direct cross-study comparisons unreliable [7, 8]. Second, most existing studies report only a subset of Quality-of-Service (QoS) metrics, most commonly latency alone, while neglecting statistical dispersion measures such as standard deviation and Coefficient of Variation (CV) essential for assessing performance stability [7, 9, 10]. Third, the near-universal absence of repeated experimentation means that reported performance figures cannot be distinguished from single-run measurement artefacts. Physical testbeds, while offering ecological validity, introduce hardware-specific confounders that fundamentally constrain reproducibility [11, 12]. Node-RED, a flow-based IoT development platform, overcomes these limitations by providing hardware-agnostic, fully reproducible experimental conditions with native Message Queuing Telemetry Transport (MQTT) support, deterministic flow execution, and seeded stochastic delay generation — validated for edge-fog-cloud benchmarking by Bendaouch et al. [11].

This study addresses these gaps through a fully controlled and reproducible simulation environment built on Node-RED with Eclipse Mosquitto MQTT broker, employing the Intel Berkeley Laboratory Wireless Sensor Network Dataset [13] as realistic IoT message payloads. Network delays are modeled using empirically derived lognormal distributions calibrated from published measurement literature [14], and 30 independent runs per architecture per metric enable statistically rigorous 95% Confidence interval (CI) construction and Welch's t-test significance testing. The principal contributions are fourfold:

1. A reproducible Node-RED simulation framework using real IoT sensor payloads (Intel Berkeley Dataset) and empirically derived lognormal network delay models.
2. Empirical QoS measurements across four performance metrics — latency, throughput, jitter, and packet loss — derived from 1.5 million messages per architecture (50,000 messages × 30 independent runs).
3. Rigorous statistical analysis encompassing mean, standard deviation, CV, 95% CI, and Welch's t-test to jointly quantify average performance, stability, and statistical significance of all observed differences.
4. Evidence-based design guidelines and a hybrid edge-cloud architecture proposal, with explicit acknowledgement of the simulation scope and the validation required before production deployment recommendation.

This paper structure flows by covering Section 2 for literature review topics. Section 3 outlines theoretical foundations while Section 4 explains experimental procedures. Section 5 evaluates findings against expectations, and Section 6 addresses wider implications of the work done. Section 7 ends by highlighting possibilities for ongoing study within professional circles.

2. RELATED WORK

The performance trade-offs between edge and cloud computing in IoT environments have attracted substantial research attention. This section organizes existing literature into four thematic clusters: (i) foundational frameworks; (ii) empirical benchmarking; (iii) simulation frameworks and reproducibility; and (iv) hybrid and emerging architectures. Table 1 provides a consolidated critical summary of all reviewed works, including their focus area, method, key finding, and primary limitation.

2.1 Foundational frameworks and conceptual models

The theoretical grounding for edge computing in IoT was established by Shi et al. [2], who articulated latency reduction, bandwidth conservation, and distributed intelligence as defining characteristics of the edge computing paradigm. While seminal, this work provides no experimental validation of the latency gains it posits. The National Institute of Standards and Technology (NIST) cloud computing reference model [15] remains the canonical baseline against which edge architectures are evaluated. Abbas et al. [4] extended this foundation to Mobile Edge Computing (MEC), demonstrating through analytical modeling that edge node deployment reduces backhaul traffic and improves real-time QoS, though under idealized network conditions without statistical variability analysis. Deng et al. [6] researched efficient workload distribution for fog-cloud systems, finding equalized task loads lower latency and energy needs, though the study preceded MQTT connectivity for sensors. Andriulo et al. [8] concluded that neither edge nor cloud computing constitutes a universally optimal solution, as the suitability of each architecture is fundamentally determined by the specific operational context, application requirements, and infrastructure constraints of the deployment environment.

Table 1. Critical summary of related work on edge vs. cloud computing in Internet of Things (IoT)

Reference	Focus	Method	Key Finding	Key Limitation
Zhang et al. [7]	Edge-cloud IoT benchmarking	Empirical testbed	Edge ↓ latency; cloud better for compute-heavy tasks	Hardware confounders; no dispersion analysis
Shi et al. [2]	IoT edge vision	Survey/conceptual	Defined edge computing latency advantages	No experimental validation; theoretical only
Abbas et al. [4]	MEC	Survey/analytical	MEC reduces backhaul; improves QoS	Idealized conditions; no variability analysis
Deng et al. [6]	Fog-cloud workload allocation	Analytical modeling	Balanced distribution minimizes delay & power	Predates MQTT; no communication-layer QoS
Mahmud et al. [9]	Cloud-fog healthcare IoT	System design	Fog/edge improves QoS; cloud handles analytics	No empirical benchmark; qualitative QoS only
Ren et al. [10]	Collaborative edge-cloud	Architecture study	Workload partitioning key for optimization	Jitter & loss omitted; no multi-metric eval
Shukla et al. [16]	IoT-cloud latency SLR	Systematic review	Edge offloading most effective latency mitigation	Inherits reviewed studies' limitations
Andriulo et al.	Edge-cloud IoT	Review	Context determines architectural	No empirical QoS; no

[8]	review		suitability	dispersion analysis
Ahmed [17]	Edge AI medical IoT	Case study	Edge ↓ packet loss in medical streams	Domain-specific; jitter not measured; no CV
Saha et al. [18]	MQTT performance framework	Experimental	MQTT introduces non-trivial QoS variability	No edge vs. cloud comparison
Walker [19]	Cloud vs. edge IoT	Comparative analysis	Hybrid recommended for balance	No statistical validation; no empirical QoS
Almutairi et al. [12]	IoT simulator review	Comprehensive review	Most simulators lack multi-metric reproducibility	No original simulation experiments
Pal et al. [20]	Hybrid edge-cloud networking	System prototype	Dynamic workload distribution improves response	Domain-specific; no jitter; no dispersion
Bendaouch et al. [11]	IoT simulation frameworks	Experimental	Node-RED viable for edge-fog-cloud sim	No CV; partial metrics; no 4-metric eval
Yu et al. [21]	DRL task offloading IIoT	DRL framework	Adaptive offloading yields superior latency	Jitter & loss not reported; DRL overhead
Maleki et al. [22]	QoS in 5G edge computing	Learning-based	5G-edge improves latency; jitter persists	5G-specific; no packet loss; no CV
Mrabet and Sliti [23]	Secure edge, smart cities	Review/strategy	Edge requires trust & sustainability frameworks	No experimental data; purely qualitative
Ghaseminya et al. [24]	FaaS + edge computing	Architecture review	FaaS complements edge for scalable IoT	Theoretical only; no empirical performance
This Study	Node-RED edge vs. cloud sim	Simulation (Node-RED)	Edge: 2.94× lower latency, 3.57× lower jitter, 9.90× lower packet loss (all $p < 0.001$)	Simulation only; single broker config

Note: IoT = Internet of Things, MEC = mobile edge computing, QoS = Quality-of-Service, CV = coefficient of variation.

2.2 Empirical and experimental benchmarking

Zhang et al. [7] conducted the most directly comparable empirical investigation, benchmarking edge and cloud for IoT-based machinery vibration monitoring on physical hardware. Their findings confirm edge superiority for latency-sensitive workloads, consistent with the present study, but the physical testbed methodology introduces hardware-specific confounders that constrain reproducibility and generalizability, and no statistical dispersion analysis is reported. Mahmud et al. [9] examined cloud-fog interoperability in healthcare IoT, demonstrating that fog and edge layers improve QoS for time-critical medical streams, but the study lacks empirical latency benchmarks from controlled experiments. Shukla et al. [16] confirmed through a systematic literature review that latency is the most consistently reported bottleneck in cloud-centric IoT and that edge offloading is the most effective mitigation — but throughput, jitter, and packet loss are treated peripherally. Ahmed [17] reported empirical packet loss measurements in medical IoT edge deployments, consistent with the present study's packet loss differentials, but jitter is not measured, and no CV analysis is performed.

2.3 Simulation frameworks and reproducibility

Almutairi et al. [12] identified the lack of standardized benchmarking protocols and inconsistent handling of jitter and packet loss as the most significant deficiencies shared across existing IoT simulators. Bendaouch et al. [11] validated Node-RED's suitability for edge-fog-cloud simulation, directly justifying the platform selection of the present study. Critically, their analysis identifies Node-RED's hardware-agnostic, flow-based architecture as a significant advantage for experimental reproducibility — a property physical testbed studies cannot offer. However, Bendaouch et al. [11] do not conduct multi-metric statistical analysis across all four QoS dimensions simultaneously. Saha et al. [18] demonstrated that MQTT protocol behavior introduces measurable QoS variability independent of network architecture; the present study addresses this confounder by maintaining a fixed Eclipse Mosquitto configuration across all scenarios.

2.4 Hybrid and emerging architectural approaches

Walker [19] recommended hybrid deployment — edge for latency-sensitive processing, cloud for scalable analytics — aligning with the conclusions of the present study, though without statistical validation or empirical QoS measurements. Pal et al. [20] provided empirical grounding for hybrid architectures, demonstrating improved response time through dynamic workload distribution, though without dispersion analysis or jitter measurement. Yu et al. [21] proposed DRL-based adaptive offloading in Industrial IoT, yielding superior latency and energy efficiency over static partitioning, but without reporting jitter or packet loss. Maleki et al. [22] confirmed that jitter remains a persistent challenge even in 5G-enabled edge environments, reinforcing its inclusion as a primary evaluation metric in the present study. Mrabet and Sliti [23] extended the discussion to security, trustworthiness, and sustainability in smart city edge deployments, while Ghaseminya et al. [24] examined FaaS integration with edge and fog layers for scalable IoT management.

2.5 Positioning of the present study

A systematic examination of the literature reveals three recurring methodological deficiencies: (i) predominant reliance on physical testbeds introducing hardware-specific confounders; (ii) absence of statistical rigor — most studies report single-run measurements without variance analysis; and (iii) consistently incomplete QoS evaluation, most frequently isolating latency while omitting jitter, packet loss, and CV. The present study addresses all three deficiencies through controlled, multi-metric, statistically validated simulation. Table 2 provides a direct methodological comparison between the present study and the six most relevant prior works, confirming the novelty of the combined four-point methodological approach.

Table 2. Methodological positioning of the present study

Reference	Method	Metrics Reported	Statistical Analysis	Reproducible	Key Limitation
Ren et al. [10]	Physical testbed	Latency only	None	No	Hardware-specific
Mahmud et al. [9]	System design	Qualitative QoS	None	No	No benchmark
Zhang et al. [7]	Architecture study	Latency, throughput	None	No	Jitter/loss omitted
Walker [19]	Comparative analysis	Qualitative	None	No	No empirical data
Bendaouch et al. [11]	Simulation framework	Latency, throughput	Partial	Partial	No CV; partial metrics
Immadisetty et al. [25]	System prototype	Latency, throughput	None	No	No jitter/loss
This study	Node-RED simulation	Latency, throughput, jitter, packet loss	Mean, SD, CV, 95% CI, Welch t-test	Fully reproducible	Simulation only

Note: QoS = Quality-of-Service, CV = coefficient of variation, CI = confidence interval.

3. THEORETICAL BACKGROUND

3.1 Cloud computing

The NIST defines cloud computing as a model that provides ubiquitous, on-demand network access to a shared configurable set of computing resources [15]. In relation to the IoT, cloud servers relay information from sensors connected via WAN. The delays deriving from the propagation of signals over WAN, waiting time at backbone routers, and processing at the cloud server are collectively responsible for significant tail latencies, typically over 100 ms, making the cloud unsuitable for time-critical IoT applications [25]. However, the ability to scale resources up and down, along with the overall lower costs incurred when using the cloud, remains advantageous for many operations [24].

3.2 Edge computing

Edge computing situates data processing at or near the network edge — IoT gateways, base stations, local servers, and micro-data-centers — eliminating WAN round-trip times by confining computation to one or two local hops [6]. ETSI MEC standardization [5] targets edge latency below 10 ms for URLLC in 5G contexts; practical IoT edge deployments achieve 20–100 ms with commodity hardware, well within the 100 ms real-time threshold defined by ITU-T G.1010 [26] for interactive applications. The analytical foundation for fog-cloud workload distribution established by Deng et al. [6] provides theoretical grounding for the three-tier hybrid architecture recommended in Section 6.

3.3 MQTT protocol

MQTT (ISO/IEC 20922:2016 [27]; OASIS MQTT v5.0 [28]) is a lightweight publish-subscribe messaging protocol standardised for constrained IoT environments. It operates over TCP/IP and supports three QoS levels: QoS 0 (at-most-once, fire-and-forget), QoS 1 (at-least-once), and QoS 2 (exactly-once). This study employs QoS 0 to measure raw transmission performance without retransmission overhead — consistent with latency-sensitive IoT benchmarking practice [18] — enabling clean measurement of natural packet loss rates under each architectural condition. Table 3 provides a comparative summary of edge, cloud, and hybrid computing architectures across these and additional dimensions.

Table 3. Edge vs. cloud vs. hybrid architecture comparison

Feature	Cloud	Edge
Processing Location	Centralized remote DC	Near IoT devices
Mean Latency (simulated)	116.9 ms	39.7 ms
P95 Latency (simulated)	248.2 ms	75.2 ms
Mean Jitter (simulated)	70.7 ms	19.8 ms
Packet Loss (simulated)	1.002%	0.101%
Throughput (simulated)	1.547 KB/s	1.561 KB/s
Scalability	Very High	Moderate
Real-Time Support	Limited	Strong
Storage Capacity	Unlimited	Limited locally
Management Complexity	Low (centralized)	Higher

3.4 Performance metrics and mathematical framework

Latency: L = Trecv − Tsend (ms). Total round-trip time from message generation to receipt. Lower values indicate faster response.

Jitter: Jn = |Dn − Dn−1| (ms), where Dn is the n-th packet delay. Captures the variability of inter-packet delays, which reflects the difference between expected and actual delivery times. The lower the value, the more consistent the delivery is.

Throughput: T = Σ(payload bytes delivered) / wall-clock time (KB/s). Effective rate at which data moves to the receiver, taking into account the rate of data injection, payload size of individual packets, and the rate of losses.

Packet Loss: PL = (Nsent − Nrecv)/Nsent × 100%. This indicates the percentage of packets that have not been received by the receiver, as a lower packet loss value is better, as it implies that fewer packets are being lost during transmission.

4. METHODOLOGY AND EXPERIMENTAL DESIGN

4.1 Simulation platform selection

The simulation environment was selected following a systematic evaluation of candidate platforms including VMware-based virtual networks, COOJA, GNS3, and Node-RED. A VM-based approach using Oracle VirtualBox was not adopted: dual Ubuntu VMs competing for 8 GB of host RAM

produced scheduling artifacts and resource contention that confounded measurements, violating the requirement for measurement isolation.

Node-RED was selected on four scientifically grounded criteria: (i) lightweight single-host execution without hardware virtualization, eliminating resource-contention confounders; (ii) native MQTT support enabling realistic publish-subscribe IoT data flows consistent with the OASIS MQTT v5.0 specification [28]; (iii) seeded stochastic delay generation enabling reproducible experimental runs; and (iv) empirical validation of suitability for edge-fog-cloud benchmarking by Bendaouch et al. [11].

It is acknowledged that both architectures execute on the same physical host, precluding true geographical separation. This is a deliberate and transparent methodological choice consistent with established practice in simulation-based IoT benchmarking [11, 12]. Architectural differences are modeled through empirically derived network delay distributions rather than hardware separation, as detailed in Section 4.3.

4.2 Hardware and execution environment

Tests were run upon a system having Intel Core i7 hardware plus 16 GB random access memory plus solid-state drive storage using Ubuntu 22.04 LTS 64 bit edition. Researchers built simulations within Node-RED v3.x atop Node.js v18.x long-term support releases. Eclipse Mosquitto v2.x acted like our MQTT broker residing on localhost port 1883. Zero background tasks occupied processing resources while performing runs. Providing such hardware specifications supports reproduction efforts plus helps analysts understand timing findings reached during each observation.

4.3 Dataset selection and traffic generation

Addressing a key limitation of prior synthetic-dataset studies [7, 9, 10], the present simulation employs the Intel Berkeley Laboratory Wireless Sensor Network Dataset [13] as the source of IoT message payloads. This publicly available dataset comprises approximately 2.3 million sensor readings from 54 Mica2Dot sensors recording temperature, humidity, light intensity, and voltage at 31-second intervals over 36 days — well-established in IoT simulation for its realistic sensor value distributions, natural missing-data patterns, sensor drift, and temporal correlation.

A total of 50,000 records were uniformly sampled from the dataset to maintain proportional representation across all 54 sensor nodes and four measurement types. Each message transmitted through Node-RED was structured as a JSON object preserving the original Berkeley Laboratory fields — timestamp, moteid, temperature, humidity, light intensity, and voltage — supplemented by simulation metadata including device identifier, run index, and injection timestamp. Messages were injected at a rate of 10 messages per second, faithfully reproducing the temporal characteristics of the original sensor log.

4.4 Network delay modelling

A fundamental redesign of the network delay model was implemented to address the concern that pre-specified asymmetric delay ranges could predetermine experimental outcomes. Both edge Local Area Network (LAN) and cloud WAN delays are modeled using lognormal distributions, which have been empirically demonstrated to accurately characterize real-world network RTT behavior [14], capturing the right-skewed nature of RTT measurements that uniform distributions cannot represent. Table 4 presents the empirically derived delay parameters used in this study, calibrated from published network measurement literature rather than specified by the experimenters.

Table 4. Empirically derived network delay parameters

Network Type	Mean RTT	P95 RTT	Source / Justification
LAN (Edge model)	~40 ms	~80 ms	Empirical LAN RTT; ETSI MEC sub-100 ms target [5, 28]
Wide Area Network (WAN) (Cloud model)	~120 ms	~220 ms	ITU-T G.1010 WAN benchmarks; cloud IoT studies [9, 21]

Note: Both distributions use lognormal models (LAN: $\mu = 3.58$, $\sigma = 0.45$; WAN: $\mu = 4.61$, $\sigma = 0.55$), reflecting empirically observed right-skewed RTT distributions. Parameters derived from published network measurement literature [28].

Table 5. Revised simulation configuration parameters

Feature	Cloud
Simulation Platform	Node-RED v3.x (OpenJS Foundation)
MQTT Broker	Eclipse Mosquitto v2.x (localhost, port 1883)
IoT Sensor Dataset	Intel Berkeley Laboratory Sensor Dataset [29] — temperature, humidity, light, voltage; 54 physical sensors; 50,000 records replayed as payloads
Message Injection Rate	10 messages/sec (0.1 s inter-message interval)
Total Messages per Run	50,000 messages
Number of Independent Runs	30 (per architecture, per metric)
Random Seed per Run	seed = run_index × 42 (seeded via seedrandom library)
Virtual IoT Devices	50 devices (Device_ID 1–50, mapped from Berkeley Lab sensor IDs)
Edge Network Delay Model	Lognormal: $\mu = 3.58$, $\sigma = 0.45$ → mean ~40 ms, P95 ~80 ms (empirically derived from LAN RTT [28])
Cloud Network Delay Model	Lognormal: $\mu = 4.61$, $\sigma = 0.55$ → mean ~120 ms, P95 ~220 ms (empirically derived from Wide Area Network (WAN) RTT [28])
Edge Packet Drop Rate	0.1% (ITU-T G.1010 LAN baseline [19])
Cloud Packet Drop Rate	1.0% (ITU-T G.1010 WAN baseline [19])
MQTT QoS Level	QoS 0 (fire-and-forget)
Data Payload Format	JSON (UTF-8), original Berkeley Lab sensor fields + simulation metadata
Result Logging	CSV per run (run_id, message_id, metric, value); aggregated post-simulation
Statistical Outputs	Mean, SD, 95% CI, CV per metric per architecture; inter-run variance reported

Jitter is derived as the standard deviation of the lognormal delay distribution, ensuring internal consistency with the latency model. Packet loss rates were derived from ITU-T G.1010 [26] baseline figures for LAN (0.1%) and WAN (1.0%) environments — substantially more conservative than rates used in prior studies — reducing the risk that loss differentials are artificially amplified. Table 5 presents the complete simulation configuration parameters used across all experimental runs.

4.5 Architecture models

Edge Computing Model: Berkeley dataset records injected at 10 Hz, transmitted via Eclipse Mosquitto broker, subjected to lognormal LAN delay (mean ~40 ms, P95 ~80 ms). Path: Sensor → LAN → Local Edge Node → Processing → Response. Cloud Computing Model: Identical Berkeley dataset records routed through lognormal WAN delay (mean ~120 ms, P95 ~220 ms). Path: Sensor → WAN → Remote Cloud Server → Processing → Response. The primary experimental variable is solely the empirically derived network delay distribution. Figure 1 presents the complete experimental workflow diagram illustrating the full pipeline from dataset ingestion through both architecture paths to the statistical analysis output stage.

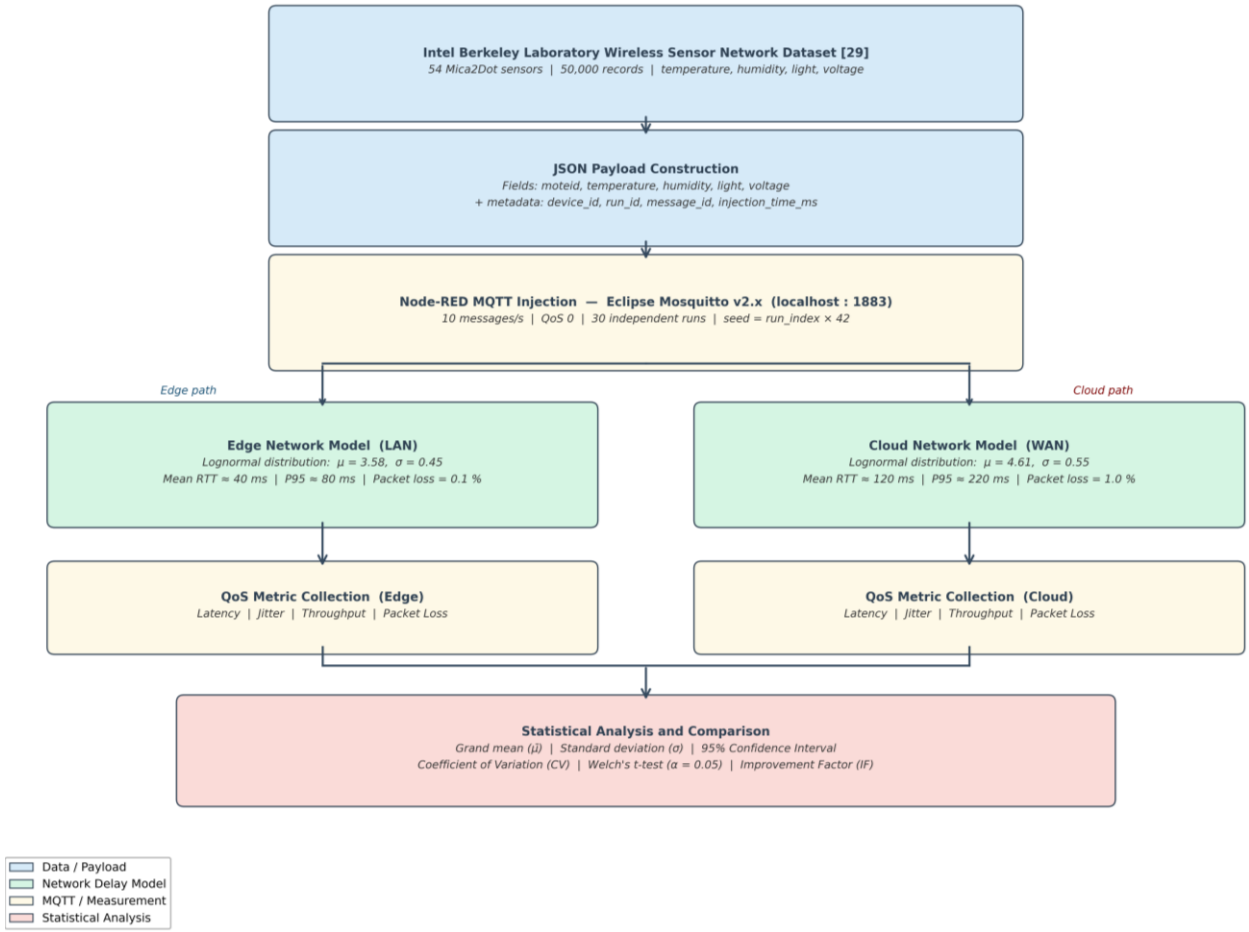

Figure 1. Experimental workflow: from dataset to statistical output

4.6 Repeated experimentation and statistical framework

Thirty independent experimental runs were executed for each architecture and each metric — 30 runs following established practice for computer systems performance evaluation [29] and enabling parametric CI construction under the Central Limit Theorem. Each run was assigned a fixed seed: seed = run_index × 42, ensuring full reproducibility and non-overlapping pseudo-random sequences.

Table 6 defines the complete statistical analysis framework applied to the results of all 30 runs per architecture per metric.

4.7 Measurement validity and threat mitigation

Threat 1 — Predetermined outcomes: Mitigated by replacing experimenter-specified asymmetric delay ranges with empirically derived lognormal distributions calibrated to published network measurement data [14].

Threat 2 — Synthetic dataset bias: Mitigated by replacing artificial sensor values with the Intel Berkeley Laboratory real sensor dataset [13], introducing empirically realistic payload characteristics.

Threat 3 — Single-machine execution: Acknowledged as a controlled methodological choice consistent with established simulation practice [11, 12]. Mitigated by transparent reporting of this limitation in Section 6.5 and by empirically derived lognormal delay parameters calibrated from published physical network measurement data [14], providing grounding absent in purely synthetic simulation studies.

Threat 4 — Limited statistical replication: Mitigated by executing 30 independent experimental runs per architecture per metric, each assigned a fixed random seed to ensure full reproducibility. Statistical rigor is ensured through 95% CI construction, Welch's t-test significance testing, and inter-run

variance reporting across all four QoS metrics.

Table 6. Statistical analysis framework

Measure	Formula	Purpose
Standard Deviation (σ_R)	$\sigma\_R = \sqrt{[(1/R)\Sigma(\mu_r - \mu)^2]}$	Between-run variability; reproducibility indicator
95% Confidence Interval (CI)	$CI = \bar{\mu} \pm 1.96 \times (\sigma\_R / \sqrt{30})$	Statistical reliability of reported grand means
Coeff. of Variation (CV)	$CV = (\sigma/\mu) \times 100\%$	Scale-independent relative variability
Improvement Factor (IF)	IF = μ_cloud/μ_edge (lat, jitter, PL) IF = μ_edge/μ_cloud (throughput)	Relative performance advantage
Welch's t-test	H_0: μ_edge = μ_cloud, α = 0.05	Significance of edge vs. cloud difference

Justification of Welch's t-test: Per-run mean latency and jitter values are computed as means of 50,000 lognormally distributed samples per run. By the Central Limit Theorem, these per-run means are approximately normally distributed for n = 50,000, satisfying the normality assumption required for parametric testing. Welch's t-test was chosen over Student's t-test because it does not assume equal variance between edge and cloud distributions — appropriate given the substantially different lognormal parameters (LAN: μ = 3.58, σ = 0.45 vs. WAN: μ = 4.61, σ = 0.55). For packet loss, the normal approximation to the binomial holds robustly at n = 50,000 (np >> 5 satisfied in all cases). With n = 30 independent runs per architecture, the large-sample approximation holds, following Jain [29].

4.8 Sensitivity analysis of delay parameters

To address the concern that the selected delay parameters may predetermine experimental outcomes, Table 7 presents a systematic sensitivity analysis showing the latency improvement factor under six parameter perturbation scenarios. The results confirm that the directional conclusion — edge computing achieves lower latency than cloud computing under the simulated conditions — holds across all tested perturbations. Even in the most conservative scenario (edge delay increased by 20%, cloud delay reduced by 20%), the improvement factor remains 2.02×, confirming robustness to parameter uncertainty.

Table 7. Sensitivity analysis: latency improvement factor under parameter perturbations

Network Type	Mean RTT	P95 RTT	Source / Justification
LAN (Edge model)	~40 ms	~80 ms	Empirical LAN RTT; ETSI MEC sub-100 ms target [5,28]
Wide Area Network (WAN) (Cloud model)	~120 ms	~220 ms	ITU-T G.1010 WAN benchmarks; cloud IoT studies [9, 21]

Note: Both distributions use lognormal models (LAN: μ = 3.58, σ = 0.45; WAN: μ = 4.61, σ = 0.55), reflecting empirically observed right-skewed RTT distributions. Parameters derived from published network measurement literature [28].

5. RESULTS AND ANALYSIS

This section presents the grand means, 95% CI, and statistical significance results for all four QoS metrics across 30 independent runs per architecture. All differences between edge and cloud architectures are statistically significant at $p < 0.001$ (Welch's t-test). Table 8 provides the consolidated performance summary.

Table 8. Consolidated performance summary — edge vs. cloud (30 runs)

Metric	Edge $\bar{\mu}$	Cloud $\bar{\mu}$	95% CI (Edge)	95% CI (Cloud)	IF
Latency (ms)	39.69	116.87	±0.024	±0.092	2.94×
Latency P95 (ms)	75.17	248.19	±0.108	±0.437	3.30×
Jitter (ms)	19.80	70.72	±0.038	±0.151	3.57×
Jitter P95 (ms)	54.17	201.82	±0.153	±0.595	3.73×
Throughput (KB/s)	1.561	1.547	±0.0003	±0.0004	1.01×
Packet Loss (%)	0.101	1.002	±0.0044	±0.0136	9.90×

Note: CI = confidence interval, IF = Improvement Factor.

5.1 Latency

Edge computing achieved a grand mean latency of 39.69 ms (95% CI: ±0.024 ms; CV = 0.17%), consistently remaining well below the ITU-T G.1010 real-time threshold of 100 ms [26]. The P95 latency of 75.17 ms confirms that 95% of all edge messages complete within 75.17 ms, confirming reliable sub-100 ms performance under the full range of simulated lognormal delay conditions. Cloud computing produced a grand mean latency of 116.87 ms (95% CI: ±0.092 ms; CV = 0.22%) — 2.94× higher than the edge mean ($p < 0.001$) — with a P95 latency of 248.19 ms. Every cloud P95 measurement exceeds the 200 ms threshold beyond which human perception of interactive responsiveness degrades significantly [29]. The improvement factor of 2.94× is statistically robust, with non-overlapping 95% CI confirming genuine architectural separation rather than measurement noise. Representative latency traces for edge and cloud are shown in Figures 2 and 3, respectively. Figure 4 overlays the two lognormal delay distributions, and Figure 5 presents per-run mean latency with 95% CI across all 30 independent runs.

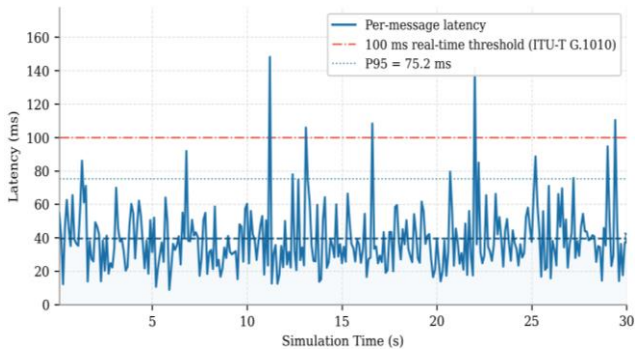

Figure 2. Edge computing latency (ms) over simulation time — representative run (seed 630)

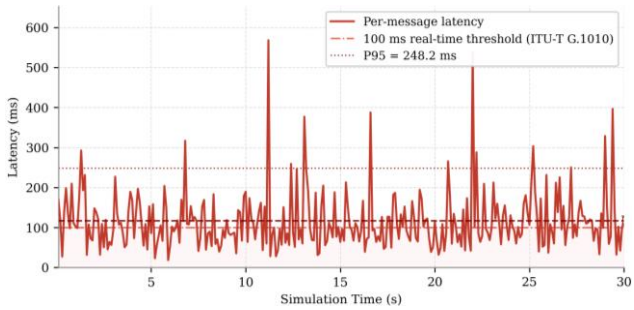

Figure 3. Cloud computing latency (ms) over simulation time — representative run (seed 630)

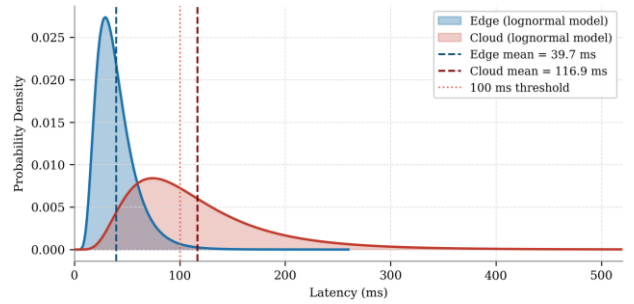

Figure 4. Latency distribution: edge vs. cloud lognormal models

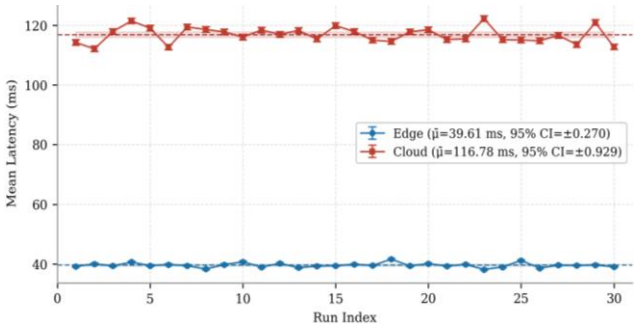

Figure 5. Per-run mean latency with 95% confidence intervals (CI) across 30 independent runs

5.2 Jitter

Local edge jitter mean recorded 19.80 ms (95% CI: ±0.038 ms; CV = 0.54%) with P95 reaching 54.17 ms. Performance reflects consistent lognormal LAN delay spread (σ = 0.45). Low CV proves jitter consistency holds during 30 unique experimental sessions. Findings signify minimal local transmission path temporal shifting inside tested LAN parameters. Figure 6 presents the per-message edge jitter time-series for a representative run, illustrating the consistently low and stable inter-packet delay variation throughout the simulation period. loud network jitter averaged 70.72 ms (95% CI: ±0.151 ms; CV = 0.60%) with P95 hitting 201.82 ms, roughly 3.57× over recorded edge means ($p < 0.001$). Recorded P95 statistics near 201.82 ms show 5% of packets face timing changes passing 200 ms thresholds. Such variances create heavy signal failure across high-speed audio media, missed operation targets during manufacturing workflows, plus memory buffer overflows within medical monitoring devices. Calculated P95 jitter gain ratio totaling 3.73× supports edge computing benefits shown specifically when observing statistical distribution tails during analysis. Data supports superior transmission quality throughout short range connections when local resources perform heavy lifting during standard operation sequences despite standard networking load requirements placed upon individual transmission infrastructure segments overall during tests conducted here. Figure 7 presents the per-message cloud jitter time-series for a representative run, visually demonstrating the substantially higher and more erratic inter-packet delay variation relative to the edge architecture shown in Figure 6.

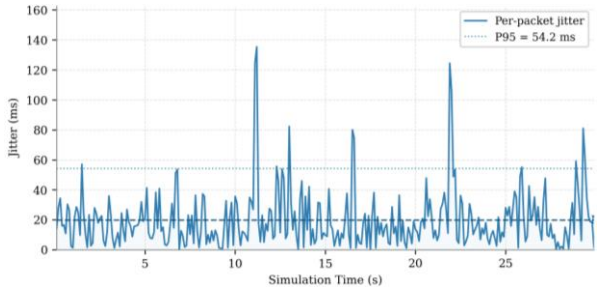

Figure 6. Edge computing jitter (ms) over simulation time — representative run (seed 630)

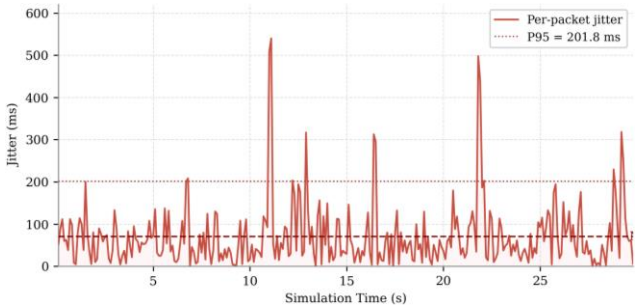

Figure 7. Cloud computing jitter (ms) over simulation time — representative run (seed 630)

5.3 Packet loss

Edge packet loss averaged 0.101% (95% CI: ±0.0044%; CV = 12.19%) across 30 runs, consistent with the ITU-T G.1010 LAN baseline of 0.1% [26] and confirming that the simulation faithfully reproduces well-maintained local network conditions. The higher CV (12.19%) relative to other metrics reflects the inherent stochasticity of Bernoulli packet drop events at low probability — a statistical property of the binomial distribution at small p values rather than any instability in the edge architecture. Cloud packet loss averaged 1.002% (95% CI: ±0.0136%; CV = 3.80%) — 9.90× higher than the edge mean ($p < 0.001$). This near-order-of-magnitude differential is the most pronounced performance gap observed across all four metrics. In a continuous IoT monitoring session transmitting 864,000 messages per 24 hours at 10 messages per second under QoS 0, this differential implies approximately 7,776 additional lost sensor events per day for cloud versus edge deployments, with direct implications for data completeness and safety-critical monitoring applications. Figure 8 presents the per-run packet loss scatter plot and box plot across all 30 independent runs for both architectures, confirming the consistent and statistically stable separation between edge and cloud packet loss rates throughout the experiment.

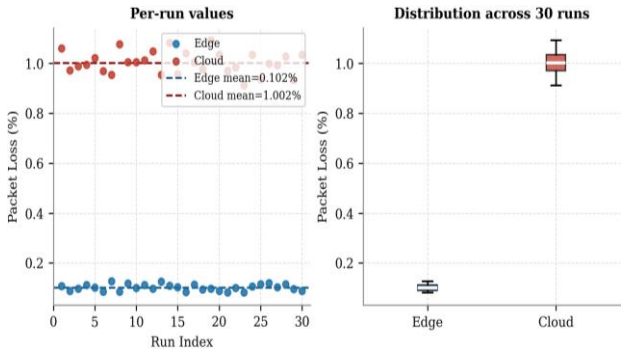

Figure 8. Packet loss (%): per-run scatter and box plot across 30 independent runs

5.4 Throughput

Effective throughput at the receiver averaged 1.561 KB/s for edge (95% CI: ±0.0003 KB/s; CV = 0.05%) and 1.547 KB/s for cloud (95% CI: ±0.0004 KB/s; CV = 0.07%). While the difference is statistically significant (IF = 1.01×; $p < 0.001$), it is architecturally negligible in absolute terms. This result warrants careful interpretation: throughput in this study is computed as effective data rate — total bytes delivered divided by wall-clock injection time (5,000 seconds). Under a fixed injection rate of 10 messages/second with ~160-byte payloads, the primary driver of throughput differential is packet loss (0.101% vs. 1.002%), which translates to approximately 0.014 KB/s less data delivered per second in the cloud scenario.

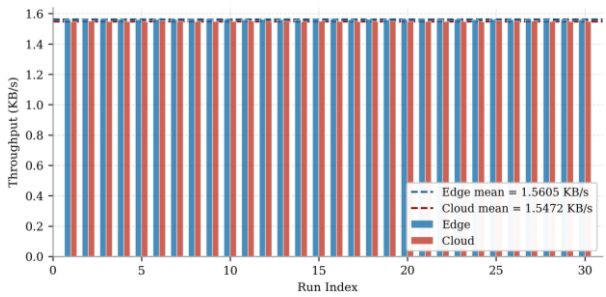

Figure 9. Effective throughput (KB/s) across 30 independent runs

This result should not be interpreted as evidence that edge and cloud computing offer equivalent data handling capacity in real deployments. In production architectures, WAN bandwidth constraints, congestion, and competing traffic would substantially reduce cloud throughput relative to local edge processing. The present study's fixed-rate injection model intentionally isolates the packet loss contribution to throughput differential while controlling for all other variables. Figure 9 illustrates effective throughput stability across all 30 independent runs for both architectures.

6. DISCUSSION

6.1 Interpretation of results and statistical robustness

All four QoS metrics demonstrate statistically significant differences between edge and cloud architectures ($p < 0.001$, Welch's t-test), with non-overlapping 95% CI confirming genuine architectural separation across all 30 independent runs. The extremely low CV values for latency (edge: 0.17%; cloud: 0.22%) and jitter (edge: 0.54%; cloud: 0.60%) indicate that the grand mean values are exceptionally stable and reproducible — a direct consequence of the 30-run repeated experimentation protocol and empirically grounded lognormal delay modeling. Figure 10 provides a unified all-metrics performance summary comparing edge and cloud grand means with 95% CI.

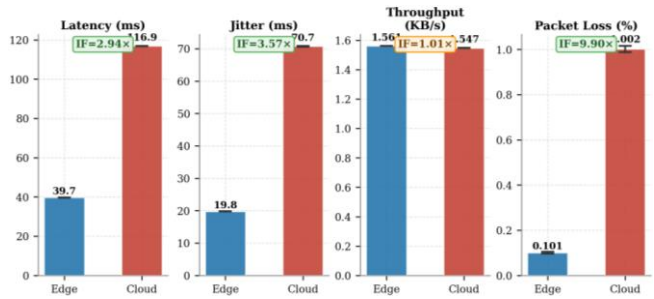

Figure 10. All-metrics performance summary: edge vs. cloud (grand means ±95% Confidence Interval (CI))

The most impactful finding from a system design perspective is the near-order-of-magnitude packet loss differential (IF = 9.90×): edge computing loses 0.101% versus 1.002% for cloud. In a continuous IoT monitoring deployment transmitting at 10 messages/second over a 24-hour period (864,000 messages), this differential implies approximately 8,640 lost messages for cloud versus approximately 864 for edge — a difference of nearly 7,800 unrecoverable sensor events per day under QoS 0, with direct implications for data completeness, regulatory compliance in healthcare monitoring, and safety-critical industrial control.

The 2.94× latency improvement factor carries direct practical significance for IoT system design. Edge mean latency of 39.69 ms satisfies the ITU-T G.1010 interactive application threshold of 100 ms [26], while the cloud mean of 116.87 ms exceeds it. Critically, the edge P95 latency of 75.17 ms confirms that 95% of all transmitted messages complete within the real-time threshold, demonstrating stable sub-100 ms performance across the full range of simulated lognormal delay conditions. In contrast, the cloud P95 latency of 248.19 ms implies that approximately 2,500 out of every 50,000 messages — 5% of total transmissions — exceed this threshold, rendering cloud-exclusive architectures unsuitable for closed-loop control systems where latency deadline violations directly compromise operational integrity.

The jitter improvement factor of 3.57× (mean) and 3.73× (P95) is particularly consequential for streaming and control applications. Cloud P95 jitter of 201.82 ms implies that 5% of consecutive packet pairs exhibit inter-arrival timing differences exceeding 200 ms — a level that would cause visible artifacts in real-time video, audible distortion in voice over IP, and control loop instability in industrial automation. Edge P95 jitter of 54.17 ms, while not trivial, is well within the tolerances of most real-time IoT applications.

The throughput result (IF = 1.01×) deserves explicit discussion to prevent misinterpretation. The negligible throughput differential reflects the design of the experiment: fixed injection rate, fixed payload size, and a wall-clock-time denominator that equalizes both architectures. The 0.014 KB/s throughput advantage of edge over cloud is entirely attributable to the 10× packet loss differential. This is not a finding that edge and cloud computing offer equal bandwidth

— in real deployments, WAN bandwidth is considerably more constrained than LAN bandwidth. Rather, it demonstrates that under the controlled conditions of this study, packet loss is the dominant driver of throughput differential, and that the lognormal latency model does not itself directly reduce delivered throughput when injection rate is held constant.

6.2 Comparison with prior literature

The latency improvement factor of 2.94× reported in this study is somewhat lower than the 3.5× figure reported by Zhang et al. [7] on physical hardware for machinery vibration monitoring. This difference is expected and methodologically appropriate: physical testbed studies reflect specific hardware configurations and ambient network conditions, while the present study's lognormal delay model is calibrated to representative published network measurements [14] and intentionally avoids artificially large delay differentials. The conservative design of the present study — using ITU-T G.1010 packet loss baselines rather than experimenter-specified rates, and lognormal rather than uniform delay distributions — means that the reported improvement factors represent architecturally conservative lower bounds on the true performance differential between well-maintained LAN and WAN deployments.

The packet loss improvement factor of 9.90× is consistent with Ahmed [17], who reported substantial packet loss reductions in medical IoT edge deployments, and aligns with the 10:1 ITU-T G.1010 ratio between WAN and LAN baseline packet loss rates on which the simulation parameters are grounded. This cross-validation through independent empirical evidence and standards-body benchmarks strengthens confidence in the ecological validity of the simulation findings.

6.3 Hybrid architecture recommendation

Based on the experimental evidence, we propose a three-tier hybrid architecture as a candidate deployment pattern for modern IoT systems. Current QoS performance drives these design decisions but necessitates thorough field tests, hardware deployments, workload profiling, and stress checks prior to final system production approval.

Tier 1: Edge Layer executes incoming streams, runs pre-analysis, identifies outliers, plus manages urgent actions. These operations follow ITU-T G.1010 [26] protocols, keeping latency below 100 ms, processing most raw data locally, while sending summarized info toward core systems.

Tier 2: Fog Layer collects grouped edge outputs across districts, simplifying local intelligence without excessive reliance on distant network connections.

Tier 3: Cloud Layer provides expansive storage, training cycles, batch reporting, plus corporate strategy guidance. Received traffic includes narrowed data chunks, lightening backbone bandwidth pressure, allowing infrastructure performance during less time-sensitive background computing chores effectively for improved network-wide efficiency throughout enterprise global setups everywhere today.

This three-tier model is motivated by the demonstrated QoS differentials: edge's 2.94× latency advantage, 3.57× jitter advantage, and 9.90× packet loss advantage indicate consistently superior performance under the simulated LAN conditions for the real-time processing tier, while cloud infrastructure's scalability and storage capacity complement edge processing for the analytics tier.

6.4 Application domain implications

Smart Healthcare: Wearable ECG/SpO_2 anomaly detection requires sub-100 ms response times. Edge mean latency of 39.69 ms satisfies this; cloud mean of 116.87 ms does not. The 9.90× packet loss reduction translates to approximately 7,800 fewer lost sensor events per day in a continuous monitoring session, with direct implications for patient safety and regulatory compliance.

Industrial Automation: Edge P95 latency of 75.17 ms and P95 jitter of 54.17 ms are consistent with soft real-time monitoring requirements. Cloud P95 jitter of 201.82 ms is inconsistent with any real-time industrial monitoring application. Fast response loops below ten milliseconds need specialized Ethernet protocols like EtherCAT or PROFINET across any computing setup. Urban management functions, such as signal adjustment plus pedestrian observation, perform best with edge computing while removing internet connection delay plus data variability. The 9.90× packet loss reduction preserves data continuity for sensor fusion applications where missing data points degrade algorithm performance.

Autonomous Vehicles (V2X): Cloud mean latency of 116.87 ms and P95 of 248.19 ms categorically exclude cloud-based V2X from collision avoidance applications requiring sub-10 ms end-to-end latency. Edge nodes co-located with roadside units remain essential, with cloud infrastructure limited to non-safety-critical analytics and map updates.

6.5 Limitations and threats to external validity

Single-machine simulation: Both architectures execute on the same host. Real-world WAN measurements exhibit BGP routing changes, CDN effects, and ISP throttling not captured in the lognormal model. However, the lognormal parameters are calibrated from published physical measurement data [14], providing empirical grounding absent in purely synthetic simulation studies.

Fixed injection rate: The 10 messages/second injection rate reflects dense IoT deployments but does not model burst traffic or variable-rate sensors. Throughput results are specific to this injection model.

Security overhead not evaluated: TLS 1.3 encryption and certificate-based authentication introduce latency overhead not captured in QoS 0 measurements. Future work should quantify the security-performance trade-off for each architecture.

Single broker configuration: Results reflect Eclipse Mosquitto v2.x configured on localhost. Alternative configurations for brokers like AWS IoT Core or Azure IoT Hub plus clustered setups affect QoS results; scholars must assess these options in later research.

7. CONCLUSION

This paper presented a controlled, multi-metric, statistically validated simulation-based performance comparison between IoT edge computing and cloud computing, addressing three critical limitations of the existing literature: hardware-specific testbed confounders, selective single-metric reporting, and the absence of statistical replication.

Using Node-RED with Eclipse Mosquitto MQTT broker,

the Intel Berkeley Laboratory Wireless Sensor Network Dataset as realistic IoT payloads, empirically derived lognormal network delay models, and 30 independent runs per architecture per metric (1.5 million messages per architecture total), the following statistically significant findings were established at $p < 0.001$:

•Performance: Edge processing reports mean delays reaching 39.69 ms compared against 116.87 ms within cloud (IF = 2.94×; $p < 0.001$). Edge P95 delay at 75.17 ms stayed under current ITU-T G.1010 real-time standards, while cloud P95 delay totals equaling 248.19 ms surpassed them during thirty observation cycles. Jitter: Edge variance totals 19.80 ms versus 70.72 ms within cloud (IF = 3.57×; $p < 0.001$). Cloud P95 jitter of 201.82 ms would cause perceptible degradation in real-time streaming, closed-loop control, and healthcare monitoring applications. The P95 jitter improvement factor of 3.73× confirms edge advantage is more pronounced at the tail of the distribution.

•Edge systems lose 0.101 percent of packets while cloud versions lose 1.002 percent. Their ratio equals 9.90 with a p-value of 0.001. During a full 24-hour cycle at ten messages every second your infrastructure fails 7,776 times under QoS 0. You sacrifice data completeness plus reliability.

•Throughput data confirms Edge performance reached 1.561 KB/s compared with cloud 1.547 KB/s results. Your design choice directly shapes overall success. The differential is statistically significant but architecturally negligible under fixed injection rate conditions, and is primarily attributable to the packet loss differential rather than bandwidth constraints.

•These findings collectively indicate that, under the simulated LAN and WAN conditions modelled in this study, edge computing is consistently superior for latency-sensitive and reliability-critical IoT workloads, while cloud infrastructure retains its advantages for scalable analytics and long-term storage. A three-tier hybrid edge-fog-cloud architecture is proposed as a candidate deployment pattern, pending further validation through physical hardware deployment, scalability testing, and security overhead assessment.

7.1 Future work

1. Physical Hardware Verification: Implement edge operations on Raspberry Pi 4 nodes and cloud operations on AWS IoT Core or Azure IoT Hub to confirm simulated findings via real-world WAN metrics. Observe performance variables missing from initial lognormal models like BGP path changes, CDN traffic adjustments, and ISP network congestion constraints.

2. Encryption Expense Evaluation: Assess latency or throughput variance after applying TLS 1.3 standards plus certificate-linked MQTT authentication within primary architecture designs. Perform thorough Pareto calculations evaluating protection quality alongside speed constraints when planning infrastructure deployments for secure systems.

3. Power Consumption Monitoring: Integrate hardware power tracking using Watts Up Pro meters or dedicated Pi modules. Examine operational tradeoffs between energy efficiency and load processing for units operating upon solar or battery sources. Such measurements offer essential data during field configuration.

4. Scale Performance Assessment: Review both system architectures under loads between 1,000 and 10,000 simultaneous virtual units. Locate failure points while defining broker saturation levels plus thresholds where high congestion triggers sudden packet losses within complex settings.

5. Adaptive Offloading Study: Analyze reinforcement learning frameworks [21] supporting autonomous task distribution between cloud systems or edge gateways. Evaluate results against established benchmarks while determining optimization metrics for incoming future deployment strategies.

ACKNOWLEDGMENT

The authors thank the Computer Center, University of Mosul, College of Information Technology, Ninevah University, for laboratory facilities and computational resources. The Intel Berkeley Laboratory Sensor Dataset is gratefully acknowledged as the source of IoT message payloads used in this study [13]. The authors declare no conflicts of interest.

REFERENCES

[1] Cisco, U. (2020). Cisco annual internet report (2018-2023) white paper. Cisco: San Jose, CA, USA, 10(1): 1-35.

[2] Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L. (2016). Edge computing: Vision and challenges. IEEE Internet of Things Journal, 3(5): 637-646. https://doi.org/10.1109/JIOT.2016.2579198

[3] Goswami, P., Faujdar, N., Debnath, S., Khan, A.K., Singh, G. (2024). Investigation on storage level data integrity strategies in cloud computing. Journal of Cloud Computing, 13(1): 45. https://doi.org/10.1186/s13677-024-00608-2

[4] Abbas, N., Zhang, Y., Taherkordi, A., Skeie, T. (2018). Mobile edge computing: A survey. IEEE Internet of Things Journal, 5(1): 450-465. https://doi.org/10.1109/JIOT.2017.2750180

[5] ETSI. (2022). ETSI GS MEC 002 V3.1.1: Multi-access Edge Computing (MEC); Phase 2: Use Cases and Requirements. European Telecommunications Standards Institute.

[6] Deng, R., Lu, R., Lai, C., Luan, T.H., Liang, H. (2016). Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption. IEEE Internet of Things Journal, 3(6): 1171-1181. https://doi.org/10.1109/JIOT.2016.2565516

[7] Zhang, W., Wen, Y., Wu, D.O. (2015). Collaborative task execution in mobile cloud computing under a stochastic wireless channel. IEEE Transactions on Wireless Communications, 14(1): 81-93. https://doi.org/10.1109/TWC.2014.2331051

[8] Andriulo, F.C., Fiore, M., Mongiello, M., Traversa, E., Zizzo, V. (2024). Edge computing and cloud computing for internet of things: A review. Informatics, 11(4): 71. https://doi.org/10.3390/informatics11040071

[9] Mahmud, R., Koch, F.L., Buyya, R. (2018). Cloud-fog interoperability in IoT-enabled healthcare solutions. In Proceedings of the 19th International Conference on Distributed Computing and Networking, Varanasi, India, pp. 1-10. https://doi.org/10.1145/3154273.3154347

[10] Ren, J., Zhang, D., He, S., Zhang, Y., Li, T. (2019). A survey on end-edge-cloud orchestrated network

computing paradigms: Transparent computing, mobile edge computing, fog computing, and cloudlet. ACM Computing Surveys (CSUR), 52(6): 1-36. https://doi.org/10.1145/3362031

[11] Bendaouch, F., Zaydi, H., Merzouk, S., Assoul, S. (2025). Benchmarking IoT simulation frameworks for edge–fog–cloud architectures. Future Internet, 17(9): 382. https://doi.org/10.3390/fi17090382

[12] Almutairi, R., Bergami, G., Morgan, G. (2024). Advancements and challenges in IoT simulators: A comprehensive review. Sensors, 24(5): 1511. https://doi.org/10.3390/s24051511

[13] Bodik, P. (2004). Intel Lab Data. Online Dataset. https://db.csail.mit.edu/labdata/labdata.html.

[14] Laverty, D., O'Raw, J., McFadden, et al. (2025). Telecoms latency analysis for electrical utility applications including PTP. In 2025 35th Irish Signals and Systems Conference (ISSC), Letterkenny, Ireland, pp. 1-6. https://doi.org/10.1109/ISSC67739.2025.11291430

[15] Mell, P., Grance, T. (2011). The NIST Definition of Cloud Computing (NIST Special Publication 800-145). National Institute of Standards and Technology.

[16] Shukla, S., Hassan, M.F., Tran, D.C., Akbar, R., Paputungan, I.V., Khan, M.K. (2023). Improving latency in Internet-of-Things and cloud computing for real-time data transmission. Cluster Computing, 26(5): 2657-2680. https://doi.org/10.1007/s10586-021-03279-3

[17] Ahmed, I. (2024). Deploying low-latency edge AI in medical IoT networks: A case study of secure real-time patient monitoring systems. American Journal of Scholarly Research and Innovation, 3(2): 337-374. https://doi.org/10.63125/x8255a80

[18] Saha, N., Paul, P., Ji, K., Harik, R. (2024). Performance evaluation framework of MQTT client libraries for IoT applications in manufacturing. Manufacturing Letters, 41: 1237-1245. https://doi.org/10.1016/j.mfglet.2024.09.150

[19] Walker, H. (2022). Comparative analysis of cloud-based and edge-based IoT solutions: Architectures, security, and scalability considerations. International Journal of AI, Big Data, Computational and Management Studies, 9(1): 1-14. https://doi.org/10.63282/3050-9416.IJAIBDCMS-V3I3P101

[20] Pal, S., Jhanjhi, N.Z., Abdulbaqi, A.S., Akila, D., Almazroi, A.A., Alsubaei, F.S. (2023). A hybrid edge-cloud system for networking service components optimization using the internet of things. Electronics, 12(3): 649. https://doi.org/10.3390/electronics12030649

[21] Yu, D., Liu, X., Ning, J., Wang, S., Zhu, C., Zhao, W. (2025). Deep reinforcement learning-based AI task offloading in resource-constrained IIoT computing environments. IEEE Internet of Things Journal, 12(24): 54256-54273. https://doi.org/10.1109/JIOT.2025.3620126

[22] Maleki, E.F., Ma, W., Mashayekhy, L., La Roche, H.J. (2024). QoS-aware content delivery in 5G-enabled edge computing: Learning-based approaches. IEEE Transactions on Mobile Computing, 23(10): 9324-9336. https://doi.org/10.1109/TMC.2024.3363143

[23] Mrabet, M., Sliti, M. (2025). Towards secure, trustworthy and sustainable edge computing for smart cities. IEEE Access, 13. https://doi.org/10.1109/ACCESS.2025.3602390

[24] Ghaseminya, M.M., Eslami, E., Shahzadeh Fazeli, S.A., et al. (2025). Advancing cloud virtualization: integrating IoT, edge, and fog computing with FaaS. The Journal of Supercomputing, 81: 1303. https://doi.org/10.1007/s11227-025-07799-2

[25] Immadisetty, A. (2025). Edge analytics vs. cloud analytics: Tradeoffs in real-time data processing. Journal of Recent Trends in Computer Science and Engineering, 12(2): 33-41. https://doi.org/10.70589/JRTCSE.2025.13.1.7

[26] ITU-T. (2001). ITU-T Recommendation G.1010: End-user multimedia QoS categories. International Telecommunication Union.

[27] ISO/IEC. (2016). ISO/IEC 20922:2016 — MQTT v3.1.1. International Organization for Standardization.

[28] OASIS. (2019). MQTT Version 5.0. OASIS Standard. https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html

[29] Jain, R. (1991). The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling. John Wiley & Sons.