ANALISIS KOMPARATIF QOS MQTT DAN COAP BERBASIS ESP32 PADA PLATFORM THINGSBOARD DALAM KONDISI JARINGAN INTERMITEN

Rohmat Nugroho, Afu Ichsan Pradana, Bondan Wahyu Pamekas
Teknik Informatika, Universitas Duta Bangsa Surakarta
Jalan Bhayangkara No. 55, Surakarta, Indonesia
rohmatnugroho99@gmail.com

ABSTRAK

Implementasi Internet of Things (IoT) di daerah terpencil Indonesia sering menghadapi kendala jaringan intermiten yang mengancam efisiensi transmisi data kritis. Penelitian ini bertujuan melakukan analisis komparatif kinerja Quality of Service (QoS) antara protokol MQTT dan CoAP menggunakan mikrokontroler ESP32 dan platform ThingsBoard. Metode penelitian melibatkan eksperimen laboratorium dengan simulasi gangguan jaringan melalui perangkat lunak clumsy. Parameter *delay, jitter, dan throughput* diukur berdasarkan standar TIPHON menggunakan Wireshark dan diolah secara otomatis melalui Google Colab. Hasil penelitian menunjukkan bahwa protokol CoAP secara konsisten unggul dalam menjaga latensi rendah dan stabilitas *jitter*, dengan nilai *delay* tetap pada kategori sangat bagus 31,30 ms bahkan dalam kondisi gangguan berat. Di sisi lain, protokol MQTT memiliki kapasitas transmisi yang jauh lebih besar dengan nilai *throughput* mencapai 4.186 Kbps kategori sangat bagus, namun mengalami degradasi performa signifikan pada parameter *delay* 533,60 ms dan *jitter* 776,40 ms akibat beban mekanisme *retransmission* pada lapisan TCP. Penelitian ini menyimpulkan adanya *trade-off fundamental* MQTT lebih optimal untuk pengiriman volume data besar, sedangkan CoAP lebih unggul dalam hal stabilitas dan respons cepat di lingkungan infrastruktur jaringan yang terbatas.

Kata kunci : *IoT, MQTT, CoAP, QoS, ESP32, ThingsBoard*

1. PENDAHULUAN

Perkembangan *Internet of Things* (IoT) di Indonesia telah menjangkau sektor-sektor krusial seperti pertanian modern dan pemantauan kesehatan di daerah terpencil. Namun, implementasi di lapangan sering kali terkendala oleh infrastruktur geografis dan keterbatasan akses internet di wilayah pedesaan [1].

Jarak yang jauh ke pusat infrastruktur data serta aksesibilitas fisik yang terbatas menciptakan lingkungan jaringan yang tidak stabil, ditandai dengan *delay* yang tinggi dan koneksi yang sering terputus secara mendadak [2].

Kondisi jaringan yang tidak menentu ini menimbulkan permasalahan serius terhadap efisiensi transmisi data pada aplikasi kritis, seperti pemantauan medis atau sensor pertanian, di mana rendahnya laju data (throughput) dapat menyebabkan hambatan dalam pengambilan keputusan. Tantangan teknis utama muncul dalam pemilihan protokol komunikasi yang tepat; protokol MQTT yang berbasis Transmission Control Protocol (TCP) menawarkan keandalan tinggi melalui mekanisme pengiriman ulang, namun lebih berat bagi sumber daya. Sebaliknya, CoAP yang menggunakan User Datagram Protocol (UDP) jauh lebih efisien untuk bandwidth terbatas namun sangat rentan terhadap penurunan throughput pada jaringan yang buruk[3].

Penelitian ini bertujuan untuk melakukan analisis komparatif kinerja *Quality of Service* (QoS) antara protokol MQTT dan CoAP guna mengidentifikasi protokol yang paling tangguh dalam menjaga integritas data pada kondisi jaringan intermiten. Rencana penyelesaian masalah dilakukan dengan memanfaatkan mikrokontroler ESP32 sebagai *end-node* yang handal [4] untuk mengirimkan data telemetri ke platform ThingsBoard [5].

Untuk mendapatkan hasil yang objektif, kinerja protokol akan dievaluasi berdasarkan standar *Telecommunications and Internet Protocol Harmonization Over Networks* (TIPHON) dengan parameter *delay*, *jitter*, dan *throughput* [6].

Simulasi gangguan jaringan seperti *lag* dan *drop* paket akan direkayasa secara terkontrol menggunakan perangkat lunak emulator clumsy guna meniru hambatan infrastruktur di dunia nyata secara akurat [7].

2. TINJAUAN PUSTAKA

Penelitian ini disusun dengan merujuk pada beberapa penelitian terdahulu yang relevan guna menghindari duplikasi serta untuk mengetahui kebaruan yang ditawarkan.

2.1. Penelitian Terdahulu

Penelitian ini merujuk pada beberapa studi terdahulu untuk memperkuat analisis. Kurniawan dkk. menemukan bahwa MQTT unggul dalam meminimalkan packet loss pada interval singkat, sementara CoAP lebih efisien untuk interval lama[5]. Nisa dkk. menggunakan standar TIPHON untuk mengevaluasi Wi-Fi kampus, yang kemudian diadaptasi dalam penelitian ini untuk menguji ketahanan protokol pada jaringan intermiten[6]. Selain itu, Rahma menyimpulkan bahwa CoAP memiliki throughput yang lebih stabil pada jaringan buruk dibandingkan MQTT[7]. Kebaruan penelitian ini terletak pada penggunaan emulator Clumsy untuk

menyimulasikan kondisi intermiten pada platform ThingsBoard.

2.2. Internet of Things (IoT)

Internet of Things (IoT) adalah konsep integrasi benda fisik yang saling berkomunikasi melalui jaringan internet untuk meningkatkan efisiensi kerja manusia, terutama pada sektor pertanian. Sistem ini mengandalkan interaksi otomatis antara perangkat fisik, konektivitas internet, dan pusat data cloud untuk menjalankan fungsinya tanpa batasan jarak[8].

2.3. ESP32

ESP32 merupakan chip utama dalam ekosistem IoT yang unggul karena integrasi modul Wi-Fi, Bluetooth, serta efisiensi konsumsi energi. Perangkat ini memiliki keandalan tinggi untuk pemantauan lingkungan secara berkelanjutan dan transmisi data secara real-time ke berbagai platform pemantauan melalui jaringan internet [9]

2.4. Protokol MQTT

Message Queue Telemetry Transport (MQTT) adalah protokol komunikasi berbasis Transmission Control Protocol (TCP) yang menggunakan mekanisme publish/subscribe melalui sebuah broker) Protokol ini sangat efektif untuk transmisi data dalam interval singkat karena kemampuannya meminimalkan tingkat packet loss serta menghasilkan nilai delay dan jitter yang rendah [5]

2.5. Protokol CoAP

Constrained Application Protocol (CoAP) dirancang khusus untuk perangkat dengan sumber daya terbatas menggunakan arsitektur Representational State Transfer (REST). Dengan menggunakan User Datagram Protocol (UDP) sebagai lapisan transport yang ringan, CoAP menawarkan efisiensi pada bandwidth terbatas, kecepatan transmisi tinggi, serta konsumsi daya yang hemat karena sifatnya yang connectionless[3].

2.6. QoS Standart THIPON

Quality of Service (QoS) merupakan metode evaluasi kinerja jaringan untuk memastikan stabilitas dan kecepatan koneksi Standar TIPHON (Telecommunications and Internet Protocol Harmonization Over Networks) digunakan sebagai pedoman penilaian objektif melalui empat parameter utama throughput, packet loss, delay, dan jitter [10] Pengukuran ini efektif untuk mengidentifikasi kemampuan jaringan dalam mendukung aktivitas digital pengguna secara maksimal.

2.7. Thingsboard

ThingsBoard adalah platform IoT open-source yang berfungsi untuk mengukur, mencatat, dan menganalisis data sensor secara real-time dan berkelanjutan. Platform ini berperan penting dalam otomatisasi sistem IoT, terutama untuk pemantauan kondisi cuaca mikro pada sektor pertanian dan kehutanan dengan tingkat akurasi yang tinggi [11].

.

2.8. Jaringan Intermiten dan Network Emulator Clumsy

Quality of Service (QoS) merupakan parameter krusial untuk mengevaluasi kinerja jaringan, terutama pada ekosistem IoT yang sering mengalami ketidakstabilan akibat faktor geografis dan keterbatasan infrastruktur fisik. Karakteristik sinyal yang lemah di area sulit, seperti pegunungan, secara langsung menurunkan kualitas komunikasi perangkat. Untuk mensimulasikan kondisi tersebut dalam lingkungan uji, digunakan perangkat lunak clumsy, yaitu emulator jaringan berbasis Windows yang mampu melakukan penurunan kualitas jaringan secara terkendali dan interaktif. Alat ini memungkinkan pemberian gangguan jaringan secara akurat tanpa mengubah kode aplikasi, sehingga memudahkan peneliti untuk mengevaluasi ketahanan protokol terhadap gangguan sinyal di lapangan secara lebih efisien [7].

.

3. METODE PENELITIAN

Penelitian tersebut dilakukan pada kondisi jaringan normal, sementara penelitian penulis berfokus pada pengujian di kondisi jaringan intermiten..

3.1. Jenis dan Sumber Data

Penelitian ini menggunakan kombinasi data primer dari pengujian teknis dan data sekunder dari studi literatur untuk menjamin keakuratan serta validitas analisis performa protokol.

3.2. Data Primer

Diperoleh melalui eksperimen pada testbed IoT menggunakan mikrokontroler ESP32 yang mensimulasikan data telemetri suhu dan kelembapan sebagai variabel kontrol agar ukuran payload tetap konsisten. Pengukuran metrik Quality of Service (QoS), yang meliputi throughput, delay, dan jitter, dilakukan berdasarkan standar TIPHON menggunakan teknik packet sniffing melalui perangkat lunak Wireshark serta diolah menggunakan Google Colab. Rekayasa gangguan jaringan dilakukan secara terkontrol menggunakan software clumsy untuk membandingkan efisiensi transmisi protokol MQTT dan CoAP secara objektif.

3.3. Data Sekunder

Merupakan data pendukung yang dikumpulkan melalui studi literatur dari jurnal ilmiah, buku teks, dan dokumen teknis terkait. Referensi mencakup spesifikasi teknis ESP32, panduan API ThingsBoard, standar regulasi TIPHON, serta penelitian relevan mengenai protokol komunikasi IoT untuk memperkuat landasan teori.

3.4. Metode Pengumpulan Data

Metode pengumpulan data dilakukan secara sistematis untuk memperoleh parameter kinerja yang akurat dan komprehensif mengenai performa protokol pada jaringan intermiten. Pendekatan ini menggabungkan pemantauan teknis operasional, pengkajian landasan teori yang relevan, serta pengujian terkontrol guna menjamin validitas analisis komparatif antara protokol MQTT dan CoAP.

3.5. Observasi

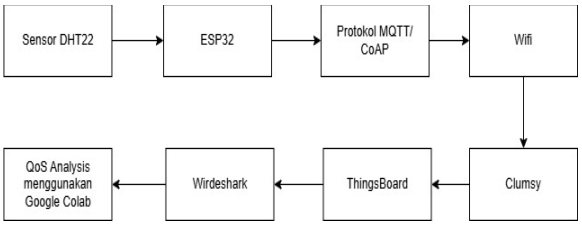
Dilakukan dengan memantau operasional sistem secara langsung melalui dashboard ThingsBoard untuk memastikan keberhasilan transmisi data telemetri, serta mengamati status koneksi pada layar LCD I2C ESP32 selama perubahan jaringan terjadi.

3.6. Studi Pustaka

Digunakan untuk memperkuat metodologi dengan mengkaji literatur terkait parameter standar kinerja jaringan IoT, struktur protokol MQTT (TCP) dan CoAP (UDP), serta konfigurasi emulator jaringan untuk mensimulasikan kondisi intermiten.

3.7. Eksperimen

Merupakan inti penelitian kuantitatif yang dilakukan di lingkungan terkontrol dengan menerapkan topologi jaringan fisik dan logika. Pengujian melibatkan stress test pengiriman pesan dari ESP32 ke broker ThingsBoard dibantu software clumsy untuk mensimulasikan gangguan drop dan lag secara akurat tanpa mengubah kode aplikasi.



Gambar 1. Arsitektur Sistem

Arsitektur sistem yang dirancang dalam penelitian ini menggunakan metode eksperimen laboratorium untuk mengevaluasi performa protokol MQTT dan CoAP dalam kondisi jaringan yang dinamis. Sebagaimana diilustrasikan pada Gambar 1, infrastruktur testbed dibagi menjadi tiga komponen utama: End-Node, Network Gateway, dan Server & Analysis Center :

a. Pada sisi hulu, perangkat ESP32 bertindak sebagai end-node yang berfungsi mengumpulkan data telemetry. Perangkat ini diprogram untuk mengirimkan data secara bergantian menggunakan dua protokol transport yang berbeda: MQTT yang berjalan di atas Transmission Control Protocol (TCP) dan CoAP yang berjalan di atas User Datagram Protocol (UDP). Penggunaan ESP32 didasarkan pada kapabilitas modul Wi-Fi internalnya yang efisien untuk transmisi data IoT.
b. Seluruh paket data dikirimkan melalui sebuah Wi-Fi Router yang bersifat dedicated. Penggunaan router khusus ini bertujuan untuk mengisolasi jalur komunikasi agar terbebas dari interferensi lalu lintas data luar, sehingga hasil pengukuran QoS murni dipengaruhi oleh karakteristik protokol dan simulasi gangguan jaringan yang diterapkan.
c. Data yang diteruskan oleh router diterima oleh laptop yang berfungsi sebagai server pusat. Sebelum data mencapai aplikasi tujuan, paket data melewati perangkat lunak Clumsy yang bekerja pada level network stack sistem operasi. Clumsy berperan penting dalam mensimulasikan kondisi jaringan intermiten dengan menginjeksikan gangguan berupa keterlambatan (lag) dan pembuangan paket (drop) secara terkontrol.
d. Setelah melalui proses simulasi gangguan, paket data diterima oleh platform ThingsBoard untuk verifikasi keberhasilan transmisi dan visualisasi telemetry. Secara bersamaan, perangkat lunak Wireshark melakukan proses packet sniffing untuk menangkap seluruh lalu lintas data mentah (raw packets). Data hasil tangkapan Wireshark tersebut kemudian diolah menggunakan Google Colab untuk menghitung parameter Quality of Service (QoS) yang meliputi delay, throughput, dan jitter sesuai dengan standar TIPHON.

Tabel 1. Gangguan jaringan sofware clumsy

Kondisi Jaringan	Lag(ms)	Drop(%)
Normal	0	0
Gangguan ringan	200	5
Gangguan sedang	350	12
Gangguan berat	500	20

Gangguan jaringan disimulasikan menggunakan perangkat lunak Clumsy dengan tiga scenario. Gangguan ringan (lag 200ms, drop 5%), Gangguan sedang (lag 350ms, drop 12%), dan Gangguan berat (lag 500ms, drop 20%).

Tabel 2. Kategori indeks TIPHON

Kategori Delay	Delay(ms)	Throughput (Kbps)	Jitter (ms)	Indeks
Sangat Bagus	<150 ms	>1.200 Kbps	0 ms	4
Bagus	150-300 ms	700 - 1.200 Kbps	0-75 ms	3
Sedang	300-450 ms	338 - 700 Kbps	75-125 ms	2
Jelek	>450 ms	0 - 338 Kbps	125-225 ms	1

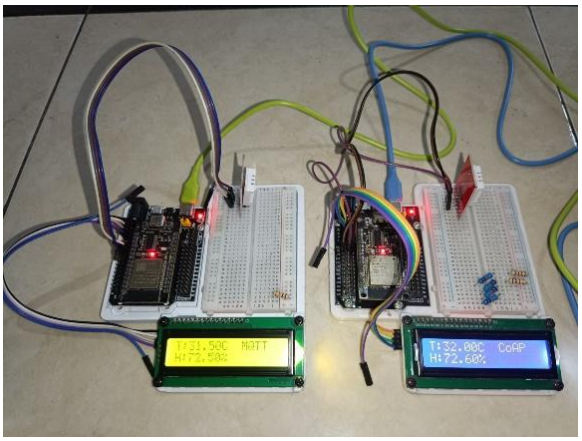
Data lalu lintas jaringan ditangkap menggunakan Wireshark untuk kemudian dihitung parameter QoS-nya berdasarkan standar TIPHON erdapat empat

tingkatan indeks kualitas yang merepresentasikan performa sistem :

a. Indeks 4 (Sangat Bagus): Merupakan kondisi ideal di mana delay berada di bawah 150 ms, laju Throughput maksimal berada di atas 1.200 Kbps, dan jitter berada pada angka 0 ms. Kategori ini menunjukkan bahwa sistem mampu melakukan transmisi data secara real-time tanpa kehilangan informasi yang berarti.
b. Indeks 3 (Bagus): Menunjukkan kualitas layanan yang masih memadai dengan rentang delay 150-300 ms, laju Throughput yang stabil pada rentang 700 - 1.200 Kbps, dan variasi keterlambatan (jitter) hingga 75 ms.
c. Indeks 2 (Sedang): Mengindikasikan adanya degradasi performa jaringan, di mana delay mencapai 300-450 ms, nilai Throughput menurun ke rentang 338 - 700 Kbps, serta nilai jitter berada di rentang 75-125 ms.
d. Indeks 1 (Jelek): Merupakan kategori terendah yang menandakan kegagalan atau hambatan serius pada jaringan. Kondisi ini terjadi apabila delay melebihi 450 ms, Throughput hanya mencapai 0 - 338 Kbps, dan jitter mencapai lebih dari 125 ms, yang berpotensi menyebabkan ketidakstabilan pada sistem IoT.

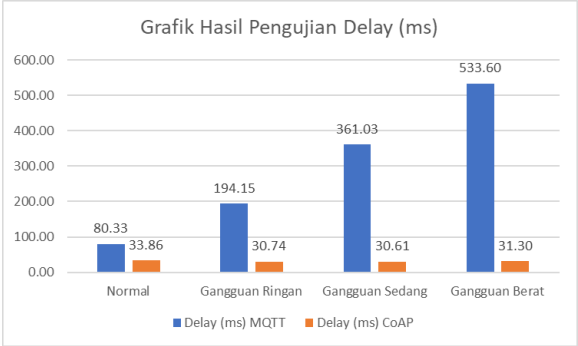
4. HASIL DAN PEMBAHASAN

Pengujian dilakukan dengan mengirimkan 500 sampel data telemetri untuk setiap kondisi jaringan pada masing-masing protokol dengan total mencapai 4000 data.



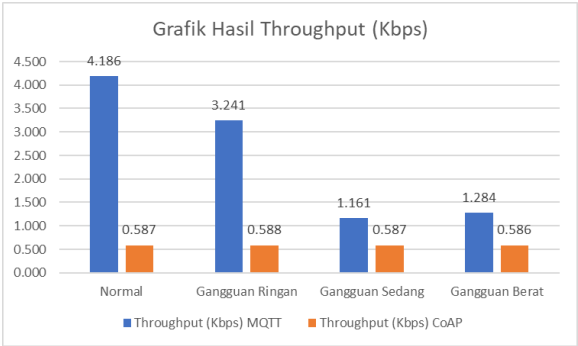
Gambar 2. Rangkaian alat pengujian

Rangkaian alat penelitian ini menggunakan mikrokontroler ESP32 sebagai pusat pemrosesan yang terhubung dengan sensor DHT22 untuk mengambil data suhu dan kelembapan melalui antarmuka digital pada pin GPIO 4. Untuk memudahkan pengawasan langsung di lapangan, sebuah layar kristal cair (LCD) yang memiliki modul I2C (Inter-Integrated Circuit) dihubungkan ke jalur bus data standar ESP32, yaitu pin GPIO 21 (SDA) dan GPIO 22 (SCL). Seluruh komponen dalam rangkaian tersebut menerima pasokan daya yang tetap dari pin 3.3V dan ground (GND) pada ESP32, sehingga menjaga kinerja sensor dan layar tetap stabil selama proses pengiriman data ke platform IoT berlangsung.



Gambar 3. Grafik hasil pengujian *delay*

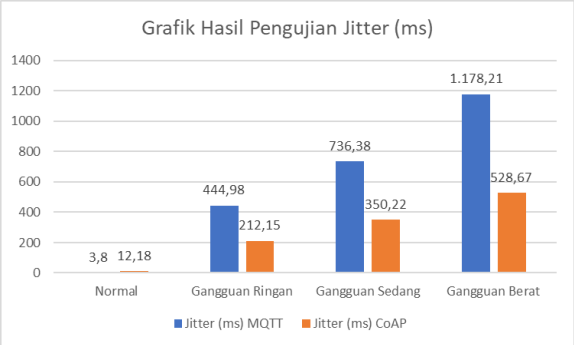
Gambar 3 Grafik delay menunjukkan tren peningkatan yang berlangsung secara linear pada protokol MQTT seiring dengan semakin meningkatnya tingkat gangguan jaringan, sementara protokol CoAP menunjukkan stabilitas yang sangat tinggi. Protokol CoAP secara terus-menerus menunjukkan nilai delay yang jauh lebih kecil dibandingkan MQTT dalam hampir semua skenario pengujian. Pada Kondisi Normal, kedua protokol berada pada kategori Sangat Bagus, namun CoAP 33,86 ms sudah menunjukkan latensi yang lebih rendah daripada MQTT 80,33 ms. Perbedaan ini menjadi semakin kontras saat gangguan jaringan diterapkan pada Gangguan Berat, MQTT mengalami lonjakan delay hingga mencapai 533,60 ms kategori Jelek, sedangkan CoAP secara impresif mampu menjaga delay tetap stabil pada angka 31,30 ms tetap kategori Sangat Bagus. Hal ini membuktikan bahwa penggunaan Google Colab untuk mengolah data mentah Wireshark berhasil menunjukkan karakteristik mendalam dari kedua protokol: mekanisme connectionless pada transport UDP yang digunakan CoAP jauh lebih efisien dalam mengirim data cepat tanpa terbebani proses handshake dan konfirmasi *retransmission* berulang, berbeda dengan protokol TCP pada MQTT yang performanya menurun drastis ketika terjadi keterlambatan atau gangguan jaringan.



Gambar 4. Grafik hasil pengujian *Throughput*

Berbeda dengan parameter delay, grafik throughput hasil pengolahan data otomatis

menggunakan Google Colab menunjukkan bahwa protokol MQTT memiliki keunggulan yang signifikan dalam hal kapasitas transmisi data dibandingkan CoAP. Meskipun jaringan mengalami manipulasi gangguan berat pada Kondisi 3, MQTT tetap mampu mempertahankan performa yang sangat baik dengan nilai throughput mencapai 1.284 Kbps, yang termasuk dalam kategori Sangat Bagus menurut standar TIPHON. Sebaliknya, CoAP menunjukkan laju data yang jauh lebih rendah namun sangat konsisten di kisaran 0,587 Kbps, yang menempatkannya secara stabil pada kategori Sedang. Fenomena ini menunjukkan bahwa mekanisme connection-oriented pada layer TCP yang digunakan MQTT lebih efektif dalam mengoptimalkan pemanfaatan bandwidth untuk menjaga volume data yang terkirim tetap tinggi. Sementara itu, mekanisme stop-and-wait yang sederhana pada tingkat aplikasi CoAP membatasi laju throughput efektifnya, meskipun protokol tersebut memberikan stabilitas pengiriman yang lebih konstan di berbagai kondisi jaringan.



Gambar 5. Grafik hasil pengujian *jitter*

Gambar 5 Grafik jitter yang diolah secara otomatis melalui Google Colab menunjukkan perbedaan waktu sampainya paket-paket yang mencerminkan tingkat stabilitas koneksi pada kedua protokol. Berdasarkan hasil pengujian, protokol CoAP menunjukkan tingkat stabilitas yang sangat impresif di mana nilai jitter tetap terjaga secara konsisten di rentang 14,48 ms hingga 16,75 ms, sehingga secara stabil berada dalam Kategori Bagus (Indeks 3) di seluruh skenario gangguan. Sebaliknya, protokol MQTT menunjukkan penurunan performa yang selaras dengan meningkatnya gangguan jaringan. Pada kondisi normal, MQTT sudah berada di kategori Sedang (Indeks 2) dengan nilai 121,40 ms, dan segera merosot ke kategori Jelek (Indeks 1) saat gangguan diterapkan, hingga mencatatkan nilai jitter terbesar sebesar 776,40 ms pada kondisi ekstrem. Fenomena ini menunjukkan adanya penumpukan paket di buffer akibat dari mekanisme pengiriman ulang (retransmission) pada lapisan TCP yang terus-menerus gagal ketika jaringan intermiten.

Hasil analisis ini mengonfirmasi bahwa CoAP jauh lebih stabil secara angka dibandingkan MQTT karena didukung oleh sifat connectionless pada UDP yang tidak memiliki beban manajemen antrean koneksi atau konfirmasi handshake yang kompleks. Dengan menggunakan Google Colab, pengolahan data mentah dari Wireshark menjadi nilai jitter ini dapat dilakukan dengan lebih akurat untuk membedakan karakteristik stabilitas antara kedua protokol tersebut sesuai dengan standarisasi TIPHON.

Tabel 3. Hasil pengujian *delay*

Kondisi Jaringan	Protokol	Delay(ms)	Indeks TIPHON	Kategori
Normal	MQTT	80.33	4	Sangat Bagus
	CoAP	33.86	4	Sangat Bagus
Gangguan Ringan	MQTT	194.15	3	Bagus
	CoAP	30.74	4	Sangat Bagus
Gangguan Sedang	MQTT	361.03	2	Sedang
	CoAP	30.61	4	Sangat Bagus
Gangguan Berat	MQTT	533.60	1	Jelek
	CoAP	31.30	4	Sangat Bagus

Hasil pengujian menunjukkan bahwa protokol CoAP secara konsisten mempertahankan tingkat latensi yang jauh lebih rendah dan stabil dibandingkan MQTT di seluruh skenario gangguan jaringan. Berdasarkan standar TIPHON, protokol CoAP mampu mempertahankan performa pada kategori "Sangat Bagus" (Indeks 4) secara kontinu dari kondisi normal hingga gangguan berat, dengan nilai delay yang terjaga stabil di kisaran 30,61 ms hingga 33,86 ms. Sebaliknya, protokol MQTT mengalami degradasi performa yang signifikan seiring meningkatnya intensitas gangguan; kategorinya menurun secara bertahap dari "Sangat Bagus" pada kondisi normal, menjadi "Bagus" pada gangguan ringan, "Sedang" pada gangguan sedang, hingga akhirnya masuk kategori "Jelek" (Indeks 1) pada kondisi gangguan berat dengan nilai delay mencapai 533,60 ms. Fenomena ini mengonfirmasi bahwa efisiensi header dan sifat connectionless pada transport UDP yang digunakan oleh protokol CoAP memberikan keunggulan kecepatan respons yang jauh lebih tangguh dalam menghadapi latensi jaringan dibandingkan protokol TCP pada MQTT yang terbebani oleh proses handshake dan konfirmasi pengiriman ulang (retransmission) secara berulang ketika terjadi gangguan.

Tabel 4. Hasil pengujian *throughput*

Kondisi Jaringan	Protokol	Throughput (Kbps)	Indeks TIPHON	Kategori
Normal	MQTT	4.186	4	Sangat Bagus
	CoAP	0.587	2	Sedang
Gangguan Ringan	MQTT	3.241	4	Sangat Bagus
	CoAP	0.588	2	Sedang
Gangguan Sedang	MQTT	1.161	3	Bagus
	CoAP	0.587	2	Sedang

Kondisi Jaringan	Protokol	Throughput (Kbps)	Indeks TIPHON	Kategori
Gangguan Berat	MQTT	1.284	2	Sedang
	CoAP	0.586	4	Sangat Bagus

Dalam hal kapasitas transmisi data (throughput), protokol MQTT menunjukkan performa volume terbaik dengan secara konsisten menjaga indeks TIPHON pada kategori Sangat Bagus pada mayoritas kondisi pengujian. Hasil pengolahan data melalui Google Colab mengungkapkan bahwa MQTT mampu mentransmisikan data hingga laju 4.186 Kbps, jauh melampaui ambang batas minimal kategori ideal. Sebaliknya, protokol CoAP menunjukkan tingkat stabilitas laju data yang sangat tinggi namun berada pada volume yang lebih rendah, secara konsisten berada pada kategori Sedang di rata – rata 0,58 Kbps pada seluruh skenario gangguan.

Keunggulan volume pada MQTT ini terletak pada mekanisme manajemen bandwidth di lapisan transport TCP yang mampu mengoptimalkan laju pengiriman data telemetri untuk memanfaatkan kapasitas jaringan secara maksimal. Berbeda dengan CoAP yang menggunakan UDP, yang meskipun memiliki laju throughput yang lebih rendah karena karakteristik payload dan header yang minimalis, protokol ini mampu mempertahankan efisiensi transmisi yang sangat stabil tanpa fluktuasi besar bahkan dalam kondisi jaringan yang paling tidak stabil. Hal ini membuktikan adanya trade-off di mana MQTT lebih unggul dalam memindahkan volume data yang besar, sementara CoAP lebih mengutamakan konsistensi laju data pada sumber daya terbatas

Tabel 4. Hasil pengujian *jitter*

Kondisi Jaringan	Protokol	Jitter (ms)	Indeks TIPHON	Kategori
Normal	MQTT	121.40	2	Sedang
	CoAP	14.98	3	Bagus
Gangguan Ringan	MQTT	244.69	1	Jelek
	CoAP	16.75	3	Bagus
Gangguan Sedang	MQTT	479.49	1	Jelek
	CoAP	14.48	3	Bagus
Gangguan Berat	MQTT	776.40	1	Jelek
	CoAP	15.40	3	Bagus

Parameter jitter menggambarkan tingkat variasi waktu kedatangan antar paket yang menunjukkan sejauh mana transmisi tetap stabil. Berdasarkan hasil pengolahan data otomatis melalui Google Colab, terlihat bahwa pada kondisi Normal, CoAP berada pada kategori Bagus (Indeks 3) sementara MQTT berada pada kategori Sedang (Indeks 2). Stabilitas MQTT menurun tajam segera setelah gangguan ringan diterapkan, merosot ke kategori Jelek (Indeks 1) dengan nilai hingga 776,40 ms pada gangguan berat. Hal ini disebabkan oleh terjadinya penumpukan paket dalam antrean buffer TCP akibat mekanisme pengiriman ulang (retransmission) yang terus-menerus gagal di lingkungan jaringan intermiten. Berbeda dengan hasil sebelumnya, protokol CoAP menunjukkan stabilitas yang jauh lebih unggul dan konsisten dengan mempertahankan kategori Bagus (Indeks 3) di seluruh skenario pengujian, mulai dari kondisi normal hingga gangguan berat (stabil di kisaran 14-16 ms). Hal ini mengonfirmasi bahwa penggunaan transport UDP yang bersifat connectionless pada CoAP memberikan keunggulan dalam menjaga konsistensi waktu pengiriman data telemetry sensor karena tidak terbebani oleh manajemen antrean atau handshake yang rumit seperti pada protokol MQTT. Penggunaan teknik manipulasi jaringan ini membuktikan bahwa stabilitas MQTT sangat rentan terhadap gangguan, sementara CoAP mampu mempertahankan performa yang optimal.

5. KESIMPULAN DAN SARAN

Penelitian ini melakukan analisis komparatif performa Quality of Service (QoS) antara protokol MQTT dan CoAP berbasis ESP32 pada platform ThingsBoard dalam kondisi jaringan intermiten dengan memfokuskan evaluasi pada parameter throughput, delay, dan jitter berdasarkan standar TIPHON. Metodologi penelitian mengintegrasikan penggunaan Google Colab untuk mengolah data mentah dari Wireshark secara otomatis guna memperoleh hasil statistik yang lebih akurat dan objektif dalam membedakan karakteristik kedua protokol tersebut. Hasil pengujian mengungkapkan bahwa protokol CoAP secara konsisten unggul dalam menjaga latensi rendah dan stabilitas jitter, di mana nilai delay tetap berada pada kategori "Sangat Bagus" (31,30 ms) karena efisiensi header dan sifat connectionless pada transport UDP.

Di sisi lain, protokol MQTT menunjukkan keunggulan kapasitas transmisi yang jauh lebih besar dengan nilai throughput mencapai 4.186 Kbps (kategori "Sangat Bagus"), namun mengalami degradasi performa signifikan pada parameter delay dan jitter hingga masuk kategori "Jelek" akibat beban mekanisme handshake serta retransmission pada lapisan TCP saat menghadapi gangguan jaringan. Secara keseluruhan, penelitian ini menyimpulkan adanya trade-off fundamental di mana MQTT merupakan pilihan tepat untuk sistem yang memprioritaskan volume data besar, sedangkan CoAP lebih optimal untuk aplikasi yang memerlukan respons cepat dan stabilitas tinggi di lingkungan infrastruktur yang terbatas.

DAFTAR PUSTAKA

[1] S. W. Prakosa, M. Faisal, Y. Adhitya, J. S. Leu, M. Köppen, and C. Avian, “Design and implementation of lora based iot scheme for Indonesian rural area,” *Electron.*, vol. 10, no. 1, pp. 1–12, 2021, doi: 10.3390/electronics10010077.

[2] E. S. Oktarina, G. Alamsyah, R. Nurhalissa, and R. F. Satria, “Transformasi Perawatan Kesehatan Ibu Hamil dengan IoT: Solusi Cerdas untuk

Pemantauan Real-Time di Daerah Terpencil," *J. Algoritm.*, vol. 22, no. 1, pp. 458–467, 2025, doi: 10.33364/algoritma/v.22-1.2290.

[3] N. A. Safitri and A. S. Priambodo, "MQTT and CoAP Communication Protocol Analysis in Internet of Things System for Strawberry Hydroponic Plants," *J. Robot. Autom. Electron. Eng.*, vol. 1, no. 1, pp. 45–56, 2023, [Online]. Available: https://journal.uny.ac.id/publications/jraee/article/view/69%0Ahttps://pdfs.semanticscholar.org/81dd/8cd9e73a635138fc7890a3800996cdf56477.pdf

[4] D. N. Bestari and A. Wibowo, "An IoT-Based Real-Time Weather Monitoring System Using Telegram Bot and Thingsboard Platform," *Int. J. Interact. Mob. Technol.*, vol. 17, no. 6, pp. 4–19, 2023, doi: 10.3991/ijim.v17i06.34129.

[5] I. K. Y. Kurniawan, I. G. Andika, I. G. M. Y. Antara, and I. G. M. Ngurah, "Analisis Quality of Service Protokol MQTT, HTTP, dan CoAP dalam Pengiriman Data ke Thingsboard," 2024, [Online]. Available: https://doi.org/10.19184/isj.v9i1.43986

[6] I. S. N. Nisa, Rahmat Miyarno Saputro, Tegar Fatwa Nugroho, and Alfirna Rizqi Lahitani, "Analisis Quality of Service (QoS) Menggunakan Standar Parameter Tiphon pada Jaringan Internet Berbasis Wi-Fi Kampus 1 Unjaya," *Teknomatika J. Inform. dan Komput.*, vol. 17, no. 1, pp. 1–9, 2024, doi: 10.30989/teknomatika.v17i1.1307.

[7] A. S. T. Rahma, "Analisis Kinerja Protokol COAP dan MQTT pada Sistem Rumah Cerdas," *Smart Comp Jurnalnya Orang Pint. Komput.*, vol. 14, no. 3, pp. 759–767, 2025, doi: 10.30591/smartcomp.v14i3.8474.

[8] I. P. Sari, A. Novita, A.-K. Al-Khowarizmi, F. Ramadhani, and A. Satria, "Pemanfaatan Internet of Things (IoT) pada Bidang Pertanian Menggunakan Arduino UnoR3," *Blend Sains J. Tek.*, vol. 2, no. 4, pp. 337–343, 2024, doi: 10.56211/blendsains.v2i4.505.

[9] M. F. Kahfi and Sujono, "Implementasi Internet of Things Berbasis ESP32 dan Blynk untuk Monitoring Mikroklimat Ruang Hunian," *J. Sains Student Res.*, vol. 4, no. 1, pp. 616–621, 2026, [Online]. Available: https://ejurnal.kampusakademik.co.id/index.php/jssr/article/view/8349

[10] T. I. Safitri *et al.*, "Analisis QoS Menggunakan Standar TIPHON pada Jaringan Wi-Fi SMK Dharma Bahari Surabaya," *RIGGS J. Artif. Intell. Digit. Bus.*, vol. 4, no. 4, pp. 2433–2443, 2025, doi: 10.31004/riggs.v4i4.3906.

[11] H. Nurwarsito and R. W. Adaby, "Pengembangan Internet Of Things (IOT) Dalam Perekaman Data Iklim Mikro Dengan Platform Thingsboard," *J. Teknol. Inf. dan Ilmu Komput.*, vol. 11, no. 6, pp. 1385–1398, 2024, doi: 10.25126/jtiik.2024118987.