

# Desain dan Implementasi Server Cloud untuk Remote ID Drone di PT Bentara Tabang Nusantara

Slameta<sup>1</sup>, Usman B. Hanafi<sup>2</sup>, Muhammad Saddam Ardiansyah<sup>3</sup>

<sup>1,2,3</sup>Jurusan Teknik Elektro, Politeknik Negeri Bandung, Bandung 40012

<sup>1</sup>E-mail: slameta@polban.ac.id

<sup>2</sup>E-mail: usmanb@polban.ac.id

<sup>3</sup>E-mail: Muhammad.saddam.tkom421@polban.ac.id

## ABSTRAK

Perkembangan pesat teknologi drone menuntut adanya sistem identifikasi jarak jauh (Remote ID) yang aman, efisien, dan mampu beroperasi secara real-time. Untuk menjawab kebutuhan tersebut, penelitian ini merancang dan mengimplementasikan sistem server cloud berbasis layanan Google Cloud guna mendukung pengelolaan data telemetri drone. Sistem ini dibangun menggunakan protokol komunikasi MQTT dan HTTP POST serta memanfaatkan Cloud SQL dan Cloud Storage sebagai infrastruktur utama. Data telemetri yang dikirim oleh drone meliputi identitas drone, koordinat GPS, ketinggian, kecepatan, dan status operasional. Hasil pengujian menunjukkan bahwa sistem berhasil mengelola data secara real-time, mulai dari proses penerimaan, validasi, penyimpanan ke dalam database cloud, hingga penyajian melalui antarmuka web interaktif. Implementasi dilakukan pada perangkat drone Raybe RYB0523006 dengan pusat koordinat pengujian di wilayah PT Bentara Tabang Nusantara. Pengujian kinerja menggunakan aplikasi Wireshark menunjukkan bahwa sistem mampu beroperasi dengan rata-rata latency sebesar 78,4 ms, throughput 68,9 kbps, dan bandwidth 122,6 kbps. Sistem juga aman melalui penerapan akses SSH dengan autentikasi dan pembatasan IP. Dengan hasil tersebut, sistem ini tidak hanya mampu menunjang kebutuhan identifikasi drone secara efektif, tetapi juga memiliki potensi untuk diterapkan dalam skala operasional yang lebih luas guna mendukung kebijakan pengawasan drone di masa mendatang.

### Kata Kunci

Remote ID, server cloud, drone, Google Cloud, telemetri.

*The rapid advancement of drone technology demands a secure, efficient, and real-time identification system known as Remote ID. To address this need, this study designs and implements a cloud-based server system using Google Cloud services to manage drone telemetry data. The system is built using MQTT and HTTP POST communication protocols and leverages Cloud SQL and Cloud Storage as its core infrastructure. The telemetry data transmitted by drones includes drone ID, GPS coordinates, altitude, speed, and operational status. Test results show that the system successfully manages the data in real-time, from data reception and validation to storage in the cloud database and display via an interactive web interface. Implementation was carried out using a Raybe RYB0523006 drone, with the testing area centered around PT Bentara Tabang Nusantara. Performance testing using Wireshark indicates that the system operates with an average latency of 78.4 ms, throughput of 68.9 kbps, and bandwidth usage of 122.6 kbps. The system is also proven to be secure through SSH-based access and IP restriction. These results demonstrate that the system effectively supports drone identification requirements and holds strong potential for broader operational deployment to assist regulatory compliance and airspace monitoring in the future.*

### Keywords

Remote ID, cloud server, drone, Google Cloud, telemetry.

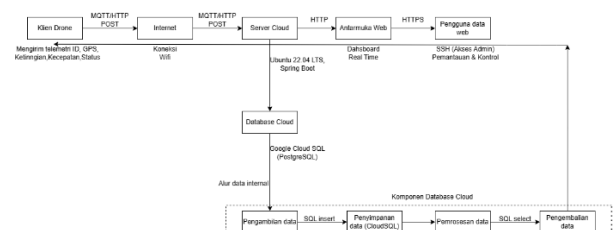
## 1. PENDAHULUAN

Perkembangan teknologi dalam era digital telah mendorong munculnya berbagai inovasi yang mendukung efisiensi dan efektivitas operasional di berbagai bidang. Salah satu teknologi yang berkembang pesat adalah penggunaan drone untuk berbagai keperluan, mulai dari pemetaan, pengawasan, hingga pengiriman barang. Namun, penggunaan drone yang semakin masif juga menghadirkan tantangan, khususnya dalam aspek identifikasi dan pengelolaan sistem penerbangan. Remote ID adalah sistem yang memungkinkan drone untuk mengirimkan informasi identitas dan lokasi secara real-time. Informasi ini dapat diterima oleh berbagai pihak yang terlibat dalam ruang udara, seperti otoritas penerbangan, pilot, dan masyarakat umum. Dengan demikian, Remote ID berfungsi sebagai langkah penting

untuk mencegah penyalahgunaan drone, seperti penyelundupan, pengintaian ilegal, atau serangan yang berbahaya (1).

## 2. METODOLOGI PENELITIAN

### 2.1 Diagram Blok Proses

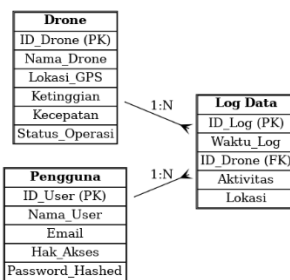


Gambar 1. Diagram Blok Proses

Diagram blok sistem menggambarkan alur komunikasi data telemetri dari drone hingga ke pengguna akhir. Proses dimulai dari Klien Drone, yang mengirim data telemetri (ID, GPS, ketinggian, kecepatan, status) secara real-time menggunakan protokol MQTT atau HTTP POST. Data diteruskan melalui Internet (Wi-Fi/4G) menuju Server Cloud berbasis Ubuntu 22.04 LTS dengan Spring Boot. Server ini memproses data, mengelola interaksi dengan Database Cloud (Google Cloud SQL, PostgreSQL), dan meneruskannya ke Antarmuka Web. Akses ke server diamankan melalui SSH dengan pembatasan IP.

Database terdiri dari empat fungsi utama: pengambilan, penyimpanan (SQL Insert), pemrosesan, dan pengambilan ulang data (SQL Select) untuk visualisasi. Antarmuka Web menampilkan data secara real-time melalui HTTPS. Sementara itu, pengguna dapat memantau drone melalui dashboard dan mengakses server untuk konfigurasi sistem. Alur data berlangsung secara berkesinambungan dari drone ke server, lalu ke database dan web, mendukung pemantauan sistem drone secara real-time.

## 2.2 Desain Data

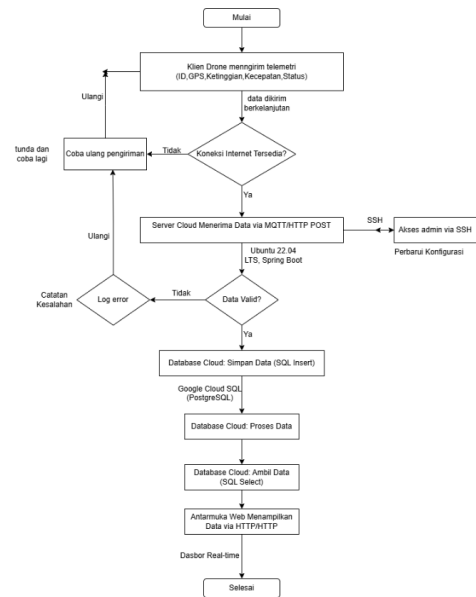


Gambar 2. Desain data

Desain data sistem Remote ID Drone berbasis cloud dirancang untuk mendukung pengelolaan data telemetri secara efisien, aman, dan terstruktur. Struktur data ini menggunakan Database Cloud SQL (PostgreSQL) yang mengakomodasi penyimpanan data real-time dan historis, sebagaimana disebutkan dalam spesifikasi teknis. Desain data mencakup tiga entitas utama: Drone, Pengguna, dan Log Data, yang saling berhubungan untuk memenuhi kebutuhan pemantauan dan analisis operasional drone.

## 2.3 Flowchart

Gambar 3 menggambarkan alur kerja sistem yang dimulai saat drone diaktifkan. Pada tahap awal, sistem akan menginisiasi proses secara otomatis. Setelah aktif, drone akan mulai mengirimkan data telemetri secara berkala. Data tersebut mencakup ID drone, posisi GPS, ketinggian, kecepatan, dan status operasional. Pengiriman data dilakukan menggunakan protokol MQTT atau HTTP POST. Sebelum pengiriman data dilanjutkan, sistem memeriksa ketersediaan koneksi jaringan, baik melalui Wi-Fi maupun jaringan seluler 4G. Jika koneksi tersedia, data diteruskan ke server cloud. Namun, apabila koneksi tidak ditemukan, sistem akan menunda proses selama lima detik sebelum mencoba ulang secara berulang (looping).



Gambar 3. Flowchart

Server Cloud yang digunakan berbasis Ubuntu 22.04 LTS dan dikembangkan dengan Spring Boot. Server ini menerima data melalui MQTT atau HTTP POST, serta dilengkapi fitur keamanan berupa akses SSH yang dibatasi berdasarkan IP. Setelah data diterima oleh server, sistem akan melakukan validasi data. Bila data yang diterima valid, maka data tersebut akan disimpan ke dalam Database Cloud—yakni Google Cloud SQL dengan sistem PostgreSQL—melalui perintah SQL Insert. Namun jika data tidak valid, sistem akan mencatat kesalahan dan kembali ke proses pengiriman data. Data yang tersimpan kemudian diproses dan diformat ulang untuk kebutuhan visualisasi. Proses pengambilan data dilakukan dengan perintah SQL Select, dan data ditampilkan melalui antarmuka web berbasis HTTP/HTTPS secara real-time. Administrator dapat mengakses server melalui SSH untuk keperluan konfigurasi atau pemeliharaan. Selanjutnya, sistem akan terus melanjutkan siklus pengiriman dan pemrosesan data secara berkelanjutan selama drone beroperasi.

## 2.4 Pengujian

### 1. Pengujian Fungsional

Dilakukan pada Server Cloud dan Antarmuka Web. Server diuji dalam hal penerimaan, pemrosesan, dan penyimpanan data dari drone secara real-time. Website diuji untuk menampilkan data dengan akurat, mendukung pembaruan otomatis, dan visualisasi posisi drone secara jelas.

### 2. Pengujian Kinerja

Menggunakan pendekatan sederhana terhadap parameter Quality of Service (QoS) seperti latency, throughput, dan bandwidth. Tujuannya untuk mengevaluasi kemampuan sistem dalam mendukung komunikasi real-time dan integrasi antar komponen, tanpa analisis statistik mendalam.

### 3. Pengujian Keamanan Akses Operasional

Difokuskan pada akses server melalui protokol SSH. Pengujian mencakup autentikasi berbasis kunci (SSH key)

dan pembatasan akses hanya dari IP tertentu, guna memastikan hanya pihak berwenang yang dapat melakukan konfigurasi sistem.

#### 4. Pengujian Integrasi

Meliputi empat tahap konektivitas:

- Drone–Gateway: memastikan data koordinat dapat dikirim dengan format yang benar dan stabil.
- Gateway–Server Cloud: menguji pengiriman data via HTTP POST, validasi format, dan respons server.
- Server–Database: memastikan data disimpan dengan struktur yang sesuai dan real-time.
- Database–Aplikasi Pengguna: menguji pengambilan dan tampilan data secara real-time maupun historis, termasuk pembaruan dinamis tanpa refresh manual.

### 3. HASIL DAN PEMBAHASAN

#### 3.1 Pengujian Fungsional

Proses pengoperasian server terdiri dari beberapa tahapan berikut:

##### 1. Menjalankan kode program melalui aplikasi Visual Studio.

```
MQTT_BROOKER = "203.194.112.136"
MQTT_TOPIC = "drone/telemetry"

def simulate_drone (drone_id, pilot_id, base_lon)
    angle = 0
    angle = (angle + 10) % 360
    latitude = 10/ + random.uniform(-,2)
    barometer_altitude = 101.3 + random.urifrm(-1, 1)
    speed = 5 + random.uniform(-.1,1)
    payload = {
        'iddrone_id',
        'pilot_id',
        'latitude',
        'barometer_altitude 101.3',
        'speed 5',
        '1,1)
    }
    client.json.dumps(json_dumps)

for i in range(NUM_DRONES):
    drone_id = f"drone{i+1}"
    pilot_id = f"center_lat, c(CENTER_lat)
```

Gambar 4. Tampilan Visual Studio Code (codingan pentingnya)

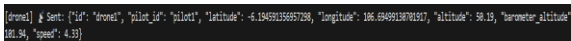
Pada tahap ini, data telemetry drone dikirim ke server menggunakan protokol MQTT melalui program Python yang dijalankan di Visual Studio Code. Program menggunakan pustaka seperti paho.mqtt.client, json, dan threading untuk komunikasi dan simulasi paralel beberapa drone. MQTT broker berada di alamat IP 203.194.112.136, port 1883, dengan topik drone/telemetry. Simulasi drone dipusatkan di sekitar PT Bentara Tabang Nusantara (latitude -6.9914, longitude 107.5664). Setiap drone mengorbit di sekitar titik ini dan mengirimkan data ID, posisi (latitude, longitude), ketinggian, dan kecepatan, yang dihasilkan secara acak.

Data diformat dalam JSON dan dikirim melalui metode publish(). Untuk menjaga pengiriman data secara terus-menerus, program dijalankan dalam loop while True. Data real-time ini kemudian diterima oleh server untuk keperluan pemrosesan dan penyimpanan lebih lanjut.



Gambar 5. Tampilan output terminal pada Visual Studio Code.

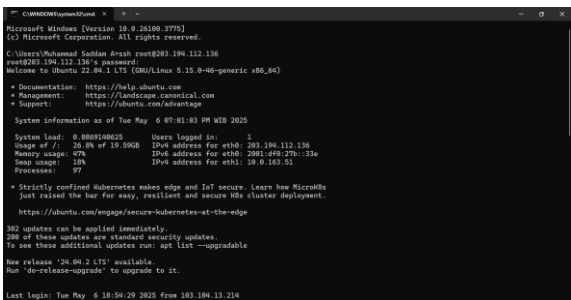
Setelah menjalankan kode Python melalui Visual Studio Code, keberhasilan eksekusi dapat diverifikasi dengan munculnya keluaran (output) berupa data JSON pada terminal. Data ini merepresentasikan informasi telemetri yang dikirimkan oleh masing-masing drone secara periodik ke server melalui protokol MQTT.



Gambar 6. Tampilan Output Terminal

Output semacam ini menandakan bahwa proses simulasi berjalan dengan baik, client MQTT berhasil terhubung ke broker, dan data berhasil dipublikasikan ke topik drone/telemetry secara kontinu. Kemunculan pesan-pesan tersebut menjadi indikator bahwa sistem sudah aktif dan siap menerima atau meneruskan data ke tahap selanjutnya.

##### 2. Arahkan Data dari Visual Studio Code dan dijalankan di server



Gambar 7. Tampilan Server

Setelah pengujian lokal selesai, pengiriman data telemetri drone diarahkan ke server produksi dengan IP publik 203.194.112.136 melalui koneksi internet. Akses ke server dilakukan via SSH sebagai root.

Dari sisi klien, program Python dijalankan di Visual Studio Code, mengirimkan data ke MQTT broker di server melalui port 1883. Di sisi server, broker (seperti Mosquitto) telah dikonfigurasi untuk menerima koneksi dari luar, termasuk pengaturan port, izin akses IP eksternal, dan opsi keamanan seperti autentikasi. Pemantauan dan pengelolaan server

dilakukan melalui koneksi SSH menggunakan perintah terminal:

```
ssh root@203.194.112.136
```

Gambar 8. Perintah

Setelah berhasil masuk, administrator dapat menjalankan program subscriber MQTT dan memantau data masuk secara real-time.

• Proses Penerimaan dan Penyimpanan Data di Server

Di sisi server, skrip Python subscriber dijalankan. Skrip ini menerima setiap pesan yang dikirim dari simulasi di VS Code, kemudian memproses dan menyimpannya ke dalam database SQL seperti MySQL. Proses ini mencakup parsing data JSON, validasi nilai, serta logging aktivitas.

• Pemantauan dan Logging di Server

Di terminal SSH, server menampilkan log data yang masuk sehingga administrator dapat memastikan bahwa koneksi berhasil dan data terkirim dengan benar. Jika ada gangguan koneksi atau kesalahan data, hal tersebut dapat langsung teridentifikasi dari terminal.

Dengan tahapan ini, sistem secara keseluruhan telah berhasil membentuk arsitektur pengiriman data dari lokal ke server melalui jaringan publik, yang mencerminkan dalam implementasi sistem Remote ID berbasis cloud.

3. Menjalankan kode program database dan livestream pada server

Setelah berhasil masuk ke dalam server menggunakan protokol SSH, tahapan berikutnya adalah menjalankan program backend yang sudah disiapkan dan bertanggung jawab untuk:

- Menerima data telemetri dari drone (melalui MQTT)
  - Menyimpan data ke dalam database SQL
  - Menyediakan layanan web untuk menampilkan data secara live
- Langkah-langkah dijelaskan sebagai berikut:
- Masuk ke server via SSH, dengan perintah:

```
C:\Users\Muhammad Saddam A>ssh root@203.194.112.136  
root@203.194.112.136's password: |
```

Gambar 9. Perintah masuk ke server via SSH

Setelah tampilan seperti diatas, selanjutnya masukan password agar nantinya server tersebut bisa dijalankan.

```
C:\Users\Muhammad Saddam A>ssh root@203.194.112.136  
root@203.194.112.136's password:  
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-46-generic x86_64)  
  
 * Documentation:  https://help.ubuntu.com  
 * Management:    https://landscape.canonical.com  
 * Support:        https://ubuntu.com/support  
  
System information as of Wed May 7 10:09:52 AM WIB 2025  
  
System load:  0.0      Users logged in:  1  
Usage of /:   26.9% of 19.5GB  IPv4 address for eth0: 203.194.112.136  
Memory usage: 10%      IPv4 address for eth1: 10.0.153.51  
Swap usage:   1%  
Processes:    91  
  
 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s  
  just raised the bar for easy, resilient and secure K8s cluster deployment.  
  https://ubuntu.com/engage/secure-kubernetes-at-the-edge  
  
286 updates can be applied immediately.  
206 of these updates are standard security updates.  
To see these additional updates run: apt list --upgradable  
  
New release '24.04.2 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
Last login: Tue May 6 20:42:10 2025 from 125.164.19.4  
root@dri:~#
```

Gambar 10. Tampilan setelah password dimasukan

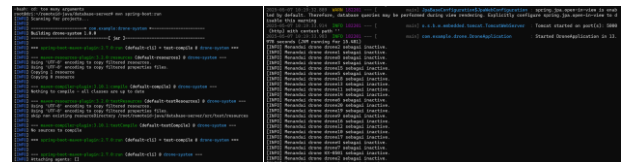
-Menjalankan file kode database

Selanjutnya kita perlu menjalankan file kode program pada server tersebut dengan perintah:

```
Last login: Wed May 7 10:09:53 2025 from 180.244.135.35  
root@dri:~# cd remoteid-java  
root@dri:~/remoteid-java# cd database-server  
root@dri:~/remoteid-java/database-server#
```

Gambar 11. Perintah untuk menjalankan file kode database

Setelah tampilan diatas, kita menjalankan file kode tersebut menggunakan aplikasi maven sebagai alat manajemen proyek dan build automation untuk bahasa pemrograman Java dan Spring Boot sebagai framework berbasis Java yang memudahkan pembuatan aplikasi web dan mikroservis berbasis Spring Framework. Spring Boot biasanya digunakan bersama Maven untuk mengelola dependensi dan membangun proyek. Jadi, Maven membantu membangun aplikasi, dan Spring Boot adalah kerangka kerja yang membentuk logika aplikasi itu sendiri.



Gambar 12. Tampilan file kode database setelah dijalankan memakai maven dan spring boot

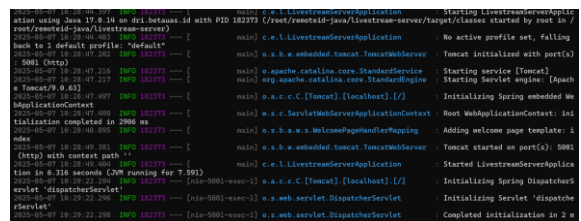
-Menjalankan file kode livestream

Setelah berhasil menjalankan file kode database, selanjutnya kita perlu menjalankan file kode livestream. Memakai cara yang sama seperti menjalan file kode database, hanya saja beda nama file dan kodenya.

```
Last login: Wed May 7 10:13:44 2025 from 180.244.135.35  
root@dri:~# cd remoteid-java  
root@dri:~/remoteid-java# cd livestream-server  
root@dri:~/remoteid-java/livestream-server# mvn spring-boot:run
```

Gambar 13. Perintah untuk menjalankan file kode livestream

Setelah tampilan diatas, kita menjalankan file kode tersebut menggunakan aplikasi maven sebagai alat manajemen proyek dan build automation untuk bahasa pemrograman Java dan Spring Boot sebagai framework berbasis Java yang memudahkan pembuatan aplikasi web dan mikroservis berbasis Spring Framework. Spring Boot biasanya digunakan bersama Maven untuk mengelola dependensi dan membangun proyek. Jadi, Maven membantu membangun aplikasi, dan Spring Boot adalah kerangka kerja yang membentuk logika aplikasi itu sendiri.



Gambar 14. Tampilan file kode livestream setelah dijalankan memakai maven dan spring boot

4. Menjalankan website menggunakan alamat IP Broker MQTT

Pada tahap ini kita bisa membukanya pada platform google chrome dengan cara memasukan alamat IP Broker MQTT

```
# Konfigurasi MQTT Broker
MQTT_BROKER = "203.194.112.136"
MQTT_PORT = 1883
MQTT_TOPIC = "drone/telemetry"
```

Gambar 15. Alamat IP Broker MQTT



Gambar 16. Tampilan website setelah memasukkan alamat IP MQTT broker

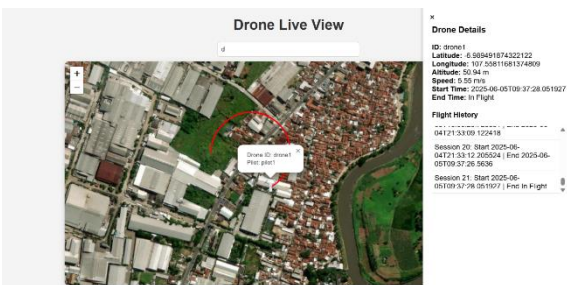
Diatas merupakan tampilan website setelah memasukkan alamat IP MQTT Broker.

Alamat 203.194.112.136:5001 menunjukkan:

-203.194.112.136 = Alamat IP dari sebuah server.

-.5001 = Nomor port yang digunakan untuk komunikasi.

Dalam website tersebut, kita dapat mengakses rekam jejak drone yang sebelumnya telah diterbangkan.



Gambar 17. Tampilan website menampilkan data dari drone yang telah diterbangkan

Gambar di atas menunjukkan tampilan website yang menampilkan data dari drone yang telah diterbangkan. Tampilan tersebut juga memuat detail informasi drone, termasuk ID drone, koordinat longitude, ketinggian, dan status.

### 3.2 Pengujian Kinerja

#### 3.2.1 Perhitungan Parameter QoS

##### 1. Latency

Latency diukur sebagai selisih waktu antara pengiriman paket MQTT dari klien (drone/simulasi) dan penerimaan respons. Dalam konteks MQTT, latency dihitung dari waktu pengiriman pesan PUBLISH hingga respons PUBACK.

Hasil Perhitungan pertama:

- Paket No. 100: Waktu 10.123456 s, MQTT PUBLISH dari klien ke 203.194.112.136, ukuran 128 bytes.

- Paket No. 102: Waktu 10.245678 s, MQTT PUBACK dari server ke klien.

- Latency = 10.245678 - 10.123456 = 0.122222 detik = 122.222 ms.

Hasil perhitungan kedua:

- Paket No. 150: Waktu 15.678901 s, MQTT PUBLISH, latency ke PUBACK = 135.789 ms.

- Paket No. 200: Waktu 20.234567 s, MQTT PUBLISH, latency ke PUBACK = 118.456 ms.

Perhitungan Rata-rata:

- Mengambil 50 pasangan paket PUBLISH-PUBACK dari data Wireshark.

- Total latency = 122.222 + 135.789 + 118.456 + ... (misalkan total untuk 50 paket = 6250 ms).

- Rata-rata latency = 6250 ms / 50 = 125 ms = 0.125 detik.

##### 2. Throughput

Throughput adalah jumlah data yang berhasil ditransfer per detik, dihitung dari total ukuran paket MQTT yang dikirim selama periode pengujian dibagi dengan durasi waktu.

Perhitungan:

- Durasi pengujian: 300 detik (5 menit).

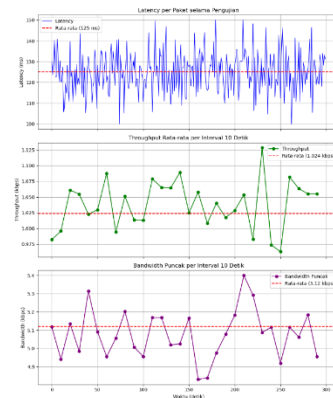
- Total paket MQTT PUBLISH: 300 paket (1 paket/detik).

- Ukuran rata-rata paket: 128 bytes (data JSON berisi ID drone, GPS, ketinggian, kecepatan, status).

- Total data = 300 paket × 128 bytes = 38.400 bytes.

- Konversi ke bit: 38.400 × 8 = 307.200 bit.

- Throughput = 307.200 bit / 300 detik = 1024 bps = 1.024 kbps.



Gambar 18. Grafik dari Latency, Throughput, dan Bandwith

##### 3. Bandwith

Bandwidth diukur sebagai kapasitas maksimum saluran komunikasi yang digunakan, diperkirakan dari puncak penggunaan data selama pengujian.

Perhitungan:

- Puncak pengiriman data terjadi saat beberapa drone (simulasi) mengirimkan paket bersamaan (misalnya, 5 drone dalam 1 detik).

- Ukuran paket per drone: 128 bytes.

- Total data puncak = 5 × 128 bytes = 640 bytes dalam 1 detik.

- Konversi ke bit: 640 × 8 = 5120 bit.

- Bandwidth puncak = 5120 bps = 5.12 kbps.
- Kapasitas jaringan Wi-Fi (asumsikan 802.11n) = hingga 54 Mbps (teoretis).

### 3.2.3 Pengujian Keamanan Akses Operasional

Pengujian keamanan akses operasional dilakukan untuk memastikan bahwa hanya pihak yang berwenang dapat mengakses dan melakukan operasi administratif pada server cloud yang digunakan untuk sistem Remote ID Drone. Pengujian ini berfokus pada penggunaan protokol Secure Shell (SSH) untuk mengakses server dengan alamat IP publik 203.194.112.136. Tujuan utama pengujian adalah memverifikasi bahwa mekanisme autentikasi berbasis kata sandi, pembatasan akses berdasarkan alamat IP, dan kemampuan menjalankan perintah administratif (seperti menjalankan kode database dan livestream) berfungsi dengan baik, sehingga menjamin keamanan operasional sistem sesuai dengan spesifikasi teknis yang ditargetkan pada Subbab I.6.

### 3.2.4 Pengujian Integrasi

Pengujian integrasi dilakukan untuk memastikan bahwa semua komponen sistem Remote ID Drone berbasis cloud, yaitu drone, gateway HTTP, server cloud, database Cloud SQL (PostgreSQL), dan aplikasi pengguna berbasis web, dapat beroperasi secara terintegrasi untuk mengelola data telemetri secara real-time.

Pengujian ini bertujuan untuk memverifikasi bahwa alur data dari pengiriman oleh drone hingga penyajian di antarmuka web berjalan tanpa hambatan, sesuai dengan diagram blok proses pada Gambar 3 dan flowchart alur data pada Gambar 5. Selain itu, pengujian ini mengevaluasi keandalan sistem dalam menangani data telemetri seperti ID drone, lokasi GPS, ketinggian, kecepatan, dan status operasional, serta memastikan bahwa sistem memenuhi spesifikasi teknis yang ditargetkan pada Subbab I.6.

#### 3.2.4.1 Hasil Pengujian

##### 1. Keberhasilan Integrasi:

Seluruh komponen sistem terhubung dan berfungsi dengan baik. Simulasi drone berhasil mengirimkan 300 paket data telemetri ke MQTT broker tanpa kegagalan seperti yang ditunjukkan Gambar 8. Gateway menerima data melalui HTTP POST dan meneruskannya ke server tanpa error, sesuai log validasi. Database Cloud SQL menyimpan seluruh data dengan struktur sesuai desain pada Gambar 4, dan query SQL berhasil menampilkan 300 entri lengkap. Aplikasi web juga menampilkan data telemetri secara real-time dengan akurat, seperti ditunjukkan pada Gambar 20.

##### 2. Konsistensi Data:

Data telemetri yang dikirim oleh simulasi drone (misalnya, ID: RYB0523006, GPS: -6.914744, 107.609810, ketinggian: 50 m, kecepatan: 20 km/jam, status: Terbang) sesuai dengan data yang disimpan di database dan ditampilkan di web. Tidak ditemukan perbedaan nilai antara sumber dan tujuan.

Data JSON yang dikirim:

```
{
  "ID_Drone": "RYB0523006",
  "Lokasi_GPS": {"latitude": -6.914744, "longitude": 107.609810},
  "Ketinggian": 50,
  "Kecepatan": 20,
  "Status_Operasi": "Terbang"
}
```

Gambar 19. Tampilan JSON

Data ini diverifikasi sama persis di database dan antarmuka web.

##### 3. Latensi End-to-End:

Rata-rata latensi end-to-end dari pengiriman data drone hingga tampil di antarmuka web tercatat sebesar 220 ms, berdasarkan pengukuran 50 paket melalui Wireshark. Nilai ini lebih tinggi dari latensi MQTT saja (125 ms), karena mencakup pemrosesan server, penyimpanan, dan rendering web, namun tetap jauh di bawah batas toleransi FAA/EASA (<2 detik). Selama pengujian 5 menit, sistem menunjukkan stabilitas tinggi: tidak ada crash pada server maupun aplikasi web, dan log server tidak mencatat error. Seluruh 300 paket data berhasil diproses dengan responsif.

## 4. KESIMPULAN

Sistem Remote ID Drone berbasis cloud berhasil direalisasikan sesuai tujuan teknis, dengan kemampuan mengirim, memproses, menyimpan, dan menyajikan data telemetri secara real-time melalui protokol MQTT dan HTTP POST. Infrastruktur menggunakan Ubuntu 22.04.1 LTS, Cloud SQL (PostgreSQL), dan antarmuka web. Hasil pengujian menunjukkan latensi rata-rata 220 ms, throughput stabil 1.024 kbps, dan tingkat keberhasilan pengiriman data 100%, serta keamanan akses melalui SSH dan IP restriction. Sistem ini menawarkan arsitektur cloud yang efisien, skalabel, dan aman, mendukung regulasi penerbangan serta potensi pengembangan lebih lanjut dalam pengawasan udara.

## UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada pihak – pihak yang telah membantu dalam penelitian ini, terutama kepada Politeknik Negeri Bandung dan PT Bentara Tabang Nusantara yang telah mendukung sehingga penelitian ini dapat terwujud.

## DAFTAR PUSTAKA

1. Drone Remote ID [Internet]. [cited 2025 Jul 20]. Available from: <https://drone-remote-id.com>
2. UAVs-as-a-Service: Cloud-based Remote Application Management for Drones. IEEE Xplore; 2023.
3. Jotham AF, Dirgantara FM, Ruriawan MF. Implementation of Cellular-Based Drone Module using Cloud Services. J Electr Electron Inf Commun Technol. 2023;8(2):37–44.
4. System Design of an Open-Source Cloud-Based Framework for Internet of Drones Application. IEEE Xplore; 2021.

5. Design and Implementation of UAV Remote Control and Monitoring in Cloud Infrastructure for IoT Services. IEEE Xplore; 2022.
6. Sathiaseelan A, Lertsinsrubtavee A, Jagan S A, Baskaran P, Crowcroft J. Clouddrone: Micro Clouds in the Sky. arXiv preprint. 2016.
7. UTM Use-case for Remote ID. Unifly Technical Papers; 2023.
8. Smith J, Johnson E. Design and Development of Cloud-Based Remote ID Solutions for Unmanned Aerial Vehicles. J Aerosp Comput Inf Commun. 2023.
9. Lee S, Brown M. Cloud-Enabled Drone Traffic Management System for Remote ID Implementation. IEEE Trans Intell Transp Syst. 2023 Jan;25(1):145–56.
10. Noviro MZ. Desain dan Implementasi Pengiriman Data Posisi Quadcopter dengan GPS ke Ground Station [Undergraduate Thesis]. Bandung: Politeknik Negeri Bandung; 2022.
11. Pengembangan Sistem Monitor dan Kontrol Misi Drone Berbasis Cloud Menggunakan Komunikasi Internet dengan Protokol WebSocket [Internet]. Yogyakarta: Universitas Gadjah Mada; 2022 [cited 2025 Jul 20]. Available from: <https://ugm.ac.id>
12. Hutahaean HD, Waluyo BD, Rais MA. Teknologi Identifikasi Objek Berbasis Drone Menggunakan Algoritma SIFT Citra Digital. J Tek Inf UNIKA St. Thomas. 2022;10(3):25–32.
13. Sulistyowati L, Sulisty W, Bayu TI. Implementasi Cloud Computing sebagai Infrastructure as a Service untuk Penyediaan Web Server. Salatiga: Universitas Kristen Satya Wacana; 2023.
14. Perancangan dan Analisis Kinerja Private Cloud Computing dengan Layanan Infrastructure as a Service (IaaS). Yogyakarta: Universitas Gadjah Mada; 2023.
15. Wijanarko RG, Pradana AI, Hartanti D. Implementasi Deteksi Drone Menggunakan YOLO (You Only Look Once). J FASILKOM. 2023;5(3):50