DOI: 10.22146/jnteti.v14i2.16272

Deteksi Duplikasi Data pada Sistem Pemantauan Kualitas Udara Berbasis IoT

Dwi Ilham Maulana[1], Asep Andang[1], Ifkar Usrah[1], Agus Purnomo[2]

[1] Jurusan Teknik Elektro, Fakultas Teknik, Universitas Siliwangi, Tasikmalaya, Jawa Barat 46115, Indonesia
[2] Jurusan Teknologi Laboratorium Medis, Poltekkes Kemenkes Tanjungkarang, Bandar Lampung, Lampung 35145, Indonesia

[Diserahkan: 19 September 2024, Direvisi: 14 Desember 2024, Diterima: 16 April 2025]
Penulis Korespondensi: Asep Andang (andhangs@unsil.ac.id)

INTISARI — Peningkatan volume data pada sistem berbasis *internet of things* (IoT) telah mendorong kebutuhan akan efisiensi dalam pengelolaan data, khususnya dalam konteks sistem pemantauan kualitas udara. Salah satu pendekatan untuk mengatasi tantangan ini adalah deteksi duplikasi data, yang berfungsi mengeliminasi data redundan guna mengurangi kebutuhan penyimpanan dan konsumsi daya. Penelitian ini bertujuan untuk mengembangkan sistem pemantauan kualitas udara berbasis IoT yang menerapkan metode deteksi duplikasi data sebagai bagian dari upaya mendukung konsep *green IoT*. Metodologi penelitian melibatkan perbandingan antara sistem tanpa dan dengan penerapan deteksi duplikasi data serta evaluasi menyeluruh terhadap kinerja sistem. Data yang diuji meliputi ukuran data yang dikirim dan konsumsi daya perangkat selama proses transmisi. Pengujian dilakukan dalam skenario operasional nyata selama 24 jam. Hasil penelitian menunjukkan bahwa penerapan deteksi duplikasi data berhasil menurunkan ukuran data yang dikirim dari 56 bita menjadi 11-44 bita, tergantung pada tingkat redundansi data. Selain itu, konsumsi daya berhasil dikurangi 1,59% hingga 3,84% dibandingkan dengan sistem tanpa deteksi duplikasi data. Metode ini juga terbukti tidak mengurangi akurasi data yang ditampilkan, sehingga tetap memenuhi kebutuhan fungsional sistem. Kesimpulannya, implementasi metode deteksi duplikasi data pada sistem pemantauan kualitas udara berbasis IoT tidak hanya mengoptimalkan proses transmisi data, tetapi juga mendukung efisiensi energi sesuai prinsip *green IoT*. Penelitian ini memberikan kontribusi penting dalam pengembangan sistem IoT yang lebih berkelanjutan dan hemat energi.

KATA KUNCI — Deteksi Data Duplikat, Deteksi Data Duplikat *Inline*, IoT, *Green IoT*, Sistem Pemantauan Kualitas Udara.

I. PENDAHULUAN

Sistem pemantauan kualitas udara adalah perangkat elektronik untuk mendeteksi kualitas udara di lingkungan, baik di dalam maupun di luar ruangan. Sebagai pemantau kualitas udara di luar ruangan, penempatan alat perlu memperhatikan kondisi meteorologi lingkungan karena berpengaruh terhadap laju penyebaran polusi udara [1]. Berdasarkan kondisi meteorologi tersebut, dimungkinkan terdapat dua atau lebih alat pemantau kualitas udara dalam satu area. Sementara itu, peningkatan alat pemantau, dalam hal ini *internet of things* (IoT), menjadi perhatian dunia, terutama dalam hal penggunaan atau konsumsi listrik untuk setiap perangkat IoT. Dengan makin berkembangnya teknologi IoT, diperkirakan pada tahun 2025 konsumsi listrik untuk perangkat IoT dapat setara dengan konsumsi listrik negara Portugal selama satu tahun, yaitu sebesar 46 TWh [2]. Meskipun sektor teknologi informasi dan komunikasi (*information and communication technologies*, ICT) bukan merupakan penyumbang besar emisi karbon dan efek rumah kaca, tetapi sektor ini menghasilkan lebih dari 2,5% dari keseluruhan emisi beracun di dunia [3]. Oleh karena itu, munculah konsep *green IoT* [4], [5].

Green IoT memungkinkan efisiensi penggunaan listrik yang lebih baik untuk setiap perangkat IoT dengan melakukan perubahan pada sisi perangkat keras dan perangkat lunak [6] sistem IoT. Di sisi perangkat lunak, perubahan dapat dilakukan pada alur pengiriman data dari pembacaan sensor ke server dengan mengurangi ukuran data yang harus dikirimkan menuju server.

Penelitian sebelumnya menunjukkan bahwa pengurangan ukuran data melalui *dynamic subsampling*, *data fusion*, dan *data scaling* mampu menurunkan ukuran data dari 96 bita menjadi 50 bita [7]. Namun, metode tersebut mengorbankan akurasi data. Hal ini menunjukkan adanya kebutuhan mendesak akan metode alternatif yang dapat mengurangi ukuran data tanpa mengorbankan akurasi.

Deteksi duplikasi data adalah salah satu metode yang menjanjikan untuk mengatasi tantangan ini. Berbeda dari metode sebelumnya, deteksi duplikasi data hanya memanipulasi potongan data yang sama tanpa memengaruhi informasi yang terkandung di dalamnya [8]. Metode ini dapat dilakukan sebelum atau setelah data disimpan. Ketika dilakukan sebelum penyimpanan, metode ini memungkinkan pengurangan ukuran data yang dikirim oleh sensor, berbeda dengan pendekatan setelah penyimpanan yang hanya memengaruhi basis data server.

Berdasarkan gap tersebut, penelitian ini bertujuan untuk membangun sistem pemantauan kualitas udara berbasis IoT yang memanfaatkan deteksi duplikasi data guna meminimalkan data duplikat yang dikirimkan secara terus-menerus. Metode ini diharapkan dapat mengurangi konsumsi energi listrik, sehingga mendukung kaidah *green IoT* [9]. Kontribusi penelitian ini adalah memberikan solusi sistem pemantauan kualitas udara yang mengintegrasikan konsep *green IoT* dengan fokus pada optimalisasi proses transmisi data melalui penerapan deteksi duplikasi data tanpa mengorbankan akurasi. Dengan dibuatnya penelitian ini, diharapkan diperoleh solusi sistem pemantauan kualitas udara yang mengimplementasikan konsep *green IoT* dengan fokus utama pada optimalisasi proses transmisi data melalui penerapan deteksi duplikasi data tanpa mengorbankan akurasi.

II. SISTEM PEMANTAUAN KUALITAS UDARA

A. PENCEMARAN UDARA

Pencemaran udara adalah masuknya atau dimasukkannya zat, energi, dan/atau komponen lain ke dalam udara ambien, sehingga udara ambien menjadi tercemar dan berkurang manfaatnya. Udara ambien merupakan udara bebas pada lapisan troposfer bumi yang dibutuhkan dan memengaruhi kesehatan manusia, makhluk hidup, dan unsur lainnya. Pencemaran diakibatkan oleh emisi yang terbentuk akibat aktivitas manusia. Maka, terdapat mutu emisi yang mengatur jumlah emisi yang boleh dibuang ke udara ambien [10].

Sebagai upaya mencegah terjadinya pencemaran udara, pemerintah membentuk baku mutu udara ambien. Baku mutu udara ambien dibagi menjadi dua, yaitu baku mutu udara ambien nasional dan baku mutu udara ambien daerah. Baku mutu udara ambien daerah ditentukan berdasarkan baku mutu udara ambien pusat serta kondisi lingkungan daerahnya. Apabila pemerintah daerah belum menetapkan baku mutu udara ambien daerahnya, maka yang berlaku adalah baku mutu udara ambien nasional.

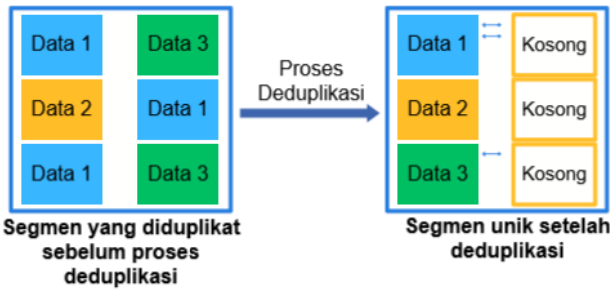
Pemerintah Indonesia melalui Stasiun Pemantau Kualitas Udara Ambien memiliki sistem pemantauan udaranya sendiri. Hasil pemantauan disajikan dalam indeks standar pencemar udara (ISPU). ISPU menggunakan status warna untuk menggambarkan kondisi mutu udara ambien di lokasi tertentu. Status ini didasarkan pada dampak terhadap kesehatan manusia, nilai estetika, dan makhluk hidup lainnya. Status warna didapatkan berdasarkan angka rentang ISPU. Parameter ISPU terdiri atas partikulat PM10 dan PM2.5, karbon monoksida (CO), nitrogen dioksida (NO_2), sulfur oksida (SO_2), ozon (O_3), dan hidrokarbon (HC) [11].

B. DETEKSI DUPLIKASI DATA

Deduplikasi data adalah metode untuk mendeteksi dan menghapus data berlebihan guna meningkatkan efisiensi penyimpanan [8], [12], seperti yang ditunjukkan pada Gambar 1. Metode ini penting dalam sistem penyimpanan skala besar karena dapat mengurangi duplikasi data, meningkatkan kapasitas penyimpanan yang tersedia, dan menurunkan biaya [13].

Secara umum, deduplikasi data dapat dibedakan berdasarkan penempatan deduplikasi, waktu deduplikasi, dan algoritma deduplikasi [14]. Berdasarkan penempatan, deduplikasi data dibagi menjadi *client based*, yaitu proses deduplikasi yang dilakukan seluruhnya di *client*; *deduplication appliance*, yaitu proses deduplikasi yang dilakukan oleh pihak ketiga; serta *storage arrays*, yaitu proses deduplikasi yang dilakukan di *server* atau tempat penyimpanan data dari *client*. Berdasarkan waktu deduplikasi, deduplikasi data terbagi menjadi *synchronous/in-band deduplication*, yaitu deduplikasi yang dilakukan sebelum data ditulis di penyimpanan data; dan *asynchronous/out-band deduplication*, yaitu deduplikasi yang dilakukan dalam kurun waktu tertentu. Berdasarkan algoritma deduplikasi, terdapat *whole file hashing*, *sub-file hashing*, dan *delta encoding*.

Proses kerja deduplikasi data biasanya terdiri atas *chunking*, *fingerprinting*, pembentukan *mega chunk*, deteksi duplikasi, pembaruan indeks, dan penyimpanan data yang unik [15]. *Chunking* adalah proses membagi data menjadi potongan-potongan data dengan ukuran tertentu. *Fingerprinting* adalah proses pembentukan nilai *hash* atau *hash signature* [14] dari *data chunk*. Pembentukan *mega chunk* adalah proses penyatuan *data chunk* yang telah melalui *fingerprinting*. *Hash signature*



Gambar 1. Proses deteksi duplikasi data.

dari *data chunk* dibandingkan dengan indeks dalam metadata yang menyimpan data unik dalam penyimpanan untuk mendeteksi duplikasi. Jika tidak ada duplikasi yang terdeteksi, berarti *data chunk* adalah unik. Data unik tersebut disimpan dalam penyimpanan dan indeks diperbarui sebagai tolok ukur untuk deteksi duplikasi berikutnya. Metadata merupakan "data-dalam-data" yang mendeskripsikan informasi spesifik [16].

Deduplikasi data adalah aspek penting dalam teknologi penyimpanan modern dan dipercaya sebagai cara efisien untuk mengoptimalkan kapasitas penyimpanan data [14]. Namun, pengembangan deduplikasi data menghadapi beberapa tantangan, seperti rasionalitas pembagian *chunk*, optimasi kinerja, kepastian reliabilitas data, skalabilitas sistem, dan penempatan data hasil deduplikasi [8]. Dibandingkan dengan metode optimalisasi penyimpanan lainnya, deduplikasi data memiliki *overhead* yang cukup besar, baik dalam proses komputasi maupun penyimpanan. Maka, pengembangan metode deduplikasi perlu terus dilakukan.

Low-overhead inline data deduplication adalah teknik deduplikasi berbasis *in-band* atau *inline deduplication* [17]. Teknik ini menggunakan dua level *fingerprinting*, yaitu *weak fingerprint* untuk mengenali duplikasi dengan cepat; dan *strong fingerprint* untuk analisis lebih akurat. Berdasarkan penelitian sebelumnya [17], *low-overhead inline data deduplication* memiliki kinerja yang lebih baik dan waktu penulisan data yang cukup dibandingkan dengan metode deduplikasi data biasa *low-overhead inline data deduplication*. Peningkatan kinerja *low-overhead inline data deduplication* dibantu oleh penggunaan *adaptive sampling deduplication detection*. *Adaptive sampling deduplication detection* adalah pembatasan sistem ketika pendeteksian data duplikasi. Apabila dalam *D*% blok data awal sistem tidak ditemukan adanya duplikasi, sistem dapat melewati blok sisanya. Namun, apabila sebelum *D*% yang ditentukan terdeteksi adanya duplikasi, proses pendeteksian duplikasi dilakukan secara menyeluruh. Hal ini membuat kinerja deduplikasi menjadi lebih cepat karena mengurangi beban kerja proses deduplikasi.

C. GREEN IOT

"*Green*" dalam *green IoT* merujuk pada karakteristik ramah lingkungan dan hemat energi, yang diterapkan baik pada perangkat keras maupun perangkat lunak. Secara sederhana, *green IoT* adalah versi IoT dengan konsumsi daya rendah, yang digagas untuk mengurangi konsumsi energi perangkat IoT seiring dengan meningkatnya penggunaan IoT secara global [9].

IoT adalah jaringan yang menghubungkan berbagai perangkat dengan identifikasi elemen yang jelas, sensor, kecerdasan tertanam, dan konektivitas internet di mana saja [18]. Ide utama IoT adalah menghubungkan berbagai hal dan memprosesnya melalui internet untuk pengendalian atau pemantauan. Seiring dengan kemajuan teknologi jaringan, IoT makin menarik perhatian karena memungkinkan setiap benda

saling berkomunikasi, terhubung, serta dikendalikan dan dipantau kapan saja serta di mana saja.

Gambar 2 memperlihatkan gambaran umum arsitektur *green IoT*. Sebuah sistem IoT, mulai dari perencanaan dan implementasi, harus mengikuti kaidah *green* [19], yang dapat diwujudkan baik dalam tingkatan perangkat lunak maupun perangkat keras. Arsitektur *green IoT* biasanya dilengkapi dengan teknologi seperti *green cloud computing*, *green radio frequency identification* (RFID), *green wireless sensor network* (WSN), *green machine-to-machine*, dan *green data center*. Meskipun demikian, belum ada arsitektur umum yang menjadi acuan dalam *green IoT*, yang membuat pengembangannya sulit untuk fokus pada aspek-aspek mendasar [6]. Oleh karena itu, diperlukan standardisasi khusus dan sebuah komite telah dibentuk untuk menetapkan standardisasi tersebut. Saat ini, komite tersebut tengah berfokus pada penentuan protokol untuk menghubungkan berbagai jenis jaringan dan perangkat yang bervariasi.

III. METODOLOGI

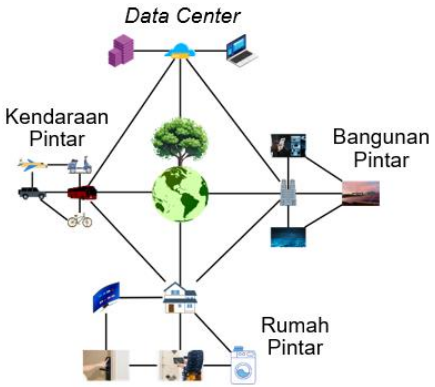
Sistem berjalan dengan penerapan metode deteksi duplikasi data yang dilakukan terhadap data yang akan dikirimkan oleh *node*. Proses dimulai dengan membagi data menjadi beberapa bagian, yang setiap bagian terdiri atas 32 karakter. Setelah pemisahan, setiap bagian diproses dengan dua langkah *fingerprinting*, yaitu *fingerprinting* pertama dan *fingerprinting* kedua.

Hasil dari *fingerprinting* pertama dibandingkan dengan data-data sebelumnya dan apabila terdapat kesamaan, dilanjutkan pada perbandingan data hasil *fingerprinting* kedua. Apabila pada perbandingan *fingerprinting* kedua ditemukan kesamaan, sistem akan mengambil data indeks yang sesuai dengan bagian tersebut. Proses perbandingan akan terus berlangsung hingga ditemukan kesamaan data, selama data sebelumnya masih ada.

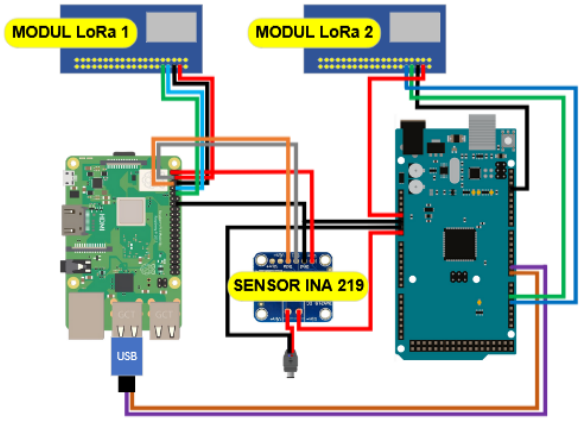
Apabila proses perbandingan selesai, langkah selanjutnya adalah menyatukan kembali bagian-bagian tersebut dengan tambahan pemisah karakter titik koma (“;”). Bagian yang memiliki kesamaan dengan data-data sebelumnya diwakili oleh indeks dari data yang memiliki kesamaan dengan bagian tersebut. Setelah bagian-bagian tersebut disatukan, data inilah yang dikirimkan menuju server.

Pada server, dilakukan pemisahan data dari *node* berdasarkan karakter pemisah. Bagian-bagian ini kemudian diperiksa, merupakan indeks server atau bukan. Apabila sebuah bagian data merupakan indeks server, sistem mengambil data asli dari basis data dan data indeks server digantikan oleh data asli. Apabila suatu bagian merupakan data asli, bagian disimpan di basis data dan diambil indeks servernya. Setelah diperiksa, semua bagian disatukan tanpa karakter pemisah, sehingga membentuk data asli dari *node*. Indeks server dari bagian yang merupakan data asli dikirimkan menuju *node*. Apabila *node* menerima data indeks server, *node* akan menyimpan data tersebut dengan bagian-bagian yang tidak memiliki kesamaan dengan data-data sebelumnya.

Sistem berjalan dengan penerapan metode deteksi duplikasi data. Prosesnya ditunjukkan pada Gambar 3. Proses deteksi duplikasi data dilakukan setelah sensor melakukan pembacaan. Hasil pembacaan setiap sensor disatukan menjadi sebuah data dengan format JavaScript Object Notation (JSON) dengan tambahan informasi *id node* yang terdaftar di basis data server. JSON merupakan sebuah format data alternatif Extensible Markup Language (XML) yang digunakan dalam pertukaran



Gambar 2. Arsitektur *green IoT*.



Gambar 3. Skematik pengujian perangkat lunak.

dan penyimpanan data. JSON terdiri atas *key* dan *value*, sehingga pertukaran data dapat dilakukan dengan mudah.

Setelah data hasil pembacaan sensor dan data *id node* disatukan dengan format JSON, data tersebut dikenai proses deteksi duplikasi. Hasil dari proses deteksi duplikasi data kemudian dikirimkan menuju server untuk disimpan dalam basis data server.

Gambar 3 merupakan alur kerja sistem deteksi duplikasi data pada sistem pemantauan kualitas udara berbasis IoT. Sistem deteksi duplikasi data pada sistem berjalan baik pada *node* dan server. Karena hal tersebut, sinkronisasi antara *node* dan server perlu diperhatikan dengan baik untuk memastikan kesamaan antara hasil pembacaan dari sensor dengan hasil pembacaan yang ditampilkan pada aplikasi atau data hasil pembacaan yang disimpan pada server.

Deteksi duplikasi data pada data JSON dilakukan dengan membagi data JSON secara keseluruhan menjadi beberapa bagian yang disebut sebagai *chunk*. Setiap *chunk* terdiri atas data dengan ukuran maksimal 32 bita. Setiap *chunk* hasil pembagian data kemudian dikenai dua kali proses *fingerprinting*. *Fingerprinting* pertama untuk setengah dari keseluruhan *data chunk* dan *fingerprinting* kedua untuk keseluruhan *data chunk*. *Fingerprinting* dilakukan menggunakan algoritma *hash* MurmurHash3. MurmurHash merupakan fungsi *hash non-cryptographic* untuk pencarian berbasis *hash* secara umum yang dibuat oleh Austin Appleby pada 2008 dengan MurmurHash3 sebagai versi terbarunya [20]. Hasil dari proses *fingerprinting* ini merupakan data-data pembanding untuk mencari data duplikat. Penggunaan MurmurHash3 ini mempercepat proses *fingerprinting* karena merupakan algoritma *hash* tercepat [20] dibandingkan dengan algoritma hash populer lain, seperti SHA-256 dan SHA-512.

Data-data pembanding merupakan data yang digunakan oleh metode deteksi duplikasi data untuk menentukan suatu data merupakan data duplikat atau bukan. Data yang dideteksi pada hal ini adalah *data chunk*. *Data chunk* yang telah diproses kemudian menjadi data *fingerprint* satu dan data *fingerprint* dua. Kedua data *fingerprint* tersebut lalu dibandingkan dengan data-data *fingerprint* dari proses sebelumnya. Apabila ada kesamaan *fingerprint*, *data chunk* yang diwakili oleh *fingerprint* tersebut merupakan data duplikat. Sebaliknya, apabila tidak terdapat kesamaan, berarti *data chunk* tersebut bukan merupakan data duplikat. Apabila dalam prosesnya terdapat dua data yang sama, tetapi salah satu data tersebut bukan merupakan data pembanding, data tersebut akan dianggap sistem bukan sebagai data duplikat. Setelah proses perbandingan selesai, *data chunk* yang terdeteksi sebagai data duplikat digantikan dengan data indeks server dan *data chunk* yang tidak terdeteksi sebagai data duplikat akan tetap sama, yaitu *data chunk* itu sendiri. Semua *data chunk* yang telah dibandingkan kemudian disatukan dalam satu daftar dengan pemisah menggunakan karakter ";" dan dikirim menuju server.

Data indeks server merupakan data angka yang didapatkan setelah *data chunk* disimpan dalam basis data. Data angka tersebut mewakili posisi *chunk* pada basis data, sehingga ketika *data chunk* tersebut dibutuhkan tinggal dilakukan pemanggilan sesuai dengan posisi *chunk* tersebut. Pada daftar yang dibuat di *node*, apabila salah satu data pada daftar tersebut merupakan *data chunk*, server akan menyimpan terlebih dahulu *data chunk* tersebut dalam basis data dan kemudian data indeks server hasil menyimpan *data chunk* disimpan untuk dikirimkan menuju *node*. Apabila terdeteksi sebagai indeks server, langkah tersebut dilewati.

Penentuan bahwa data dalam daftar merupakan *data chunk* dan data indeks server dilakukan menggunakan tipe datanya. Jika tipe datanya adalah *string*, berarti data tersebut merupakan *data chunk*, sedangkan jika tipe datanya adalah *integer*, maka data tersebut merupakan data indeks server. Setelah semua *data chunk* yang terdeteksi disimpan pada basis data, proses dilanjutkan dengan menggantikan data indeks server dengan *data chunk*-nya, sehingga semua data pada daftar yang dikirimkan oleh *node* kini merupakan *data chunk* seluruhnya.

Daftar data yang telah menjadi *data chunk* seluruhnya tersebut kemudian disatukan, sehingga menjadi data pembacaan sensor terbaru. Data pembacaan sensor terbaru di sini adalah data JSON yang dibuat sebelumnya oleh *node* sebelum melalui proses pendeteksian duplikasi data.

Setelah data pembacaan sensor terbaru disimpan, data-data indeks *server* baru dikirimkan menuju *node* dalam bentuk daftar dengan pemisah karakter "," dan diakhiri oleh karakter ";". Data-data indeks server tersebut disimpan pada *node* sebagai data pembanding baru dan digabungkan dengan data *fingerprint* satu dan *fingerprint* dua yang berkaitan.

Saat data dikirimkan dari *node* menuju server dan dari server menuju *node*, terdapat *gateway* sebagai pembantu komunikasi antara *node* dan server tersebut. Hal tersebut diperlukan karena pada *node* komunikasi hanya dapat dilakukan melalui komunikasi LoRa. Komunikasi LoRa ini tidak dapat langsung terhubung dengan jaringan internet, tetapi harus melalui perantara berupa *gateway*. *Gateway* pada sistem mengirimkan data menuju server menggunakan protokol *message queuing telemetry transport* (MQTT).

Pada aplikasi Android, data pembacaan sensor terbaru didapatkan dengan melakukan permintaan Hypertext Transfer Protocol (HTTP) secara berkala menuju server. Dari data tersebut kemudian dilakukan pemisahan untuk setiap datanya menjadi data PM 1,0, PM 2,5, PM 10, suhu, kelembapan udara, dan kecepatan angin. Data pembacaan sensor terbaru yang diterima oleh aplikasi Android adalah data hasil pembacaan sensor terbaru sesuai dengan *id* perangkat IoT yang dibutuhkan oleh aplikasi Android.

Data hasil pembacaan sensor terbaru terbagi menjadi data *id* perangkat IoT dan data pembacaan sensor. Data pembacaan sensor terdiri atas dua bagian yang dipisahkan oleh karakter "|". Bagian pertama merupakan bagian yang berisi label mengenai data hasil pembacaan sensor-sensor yang telah disatukan. Data label berbentuk karakter "h", "t", "1", "2", "0", dan "v". Label "h" mewakili data kelembapan udara, label "t" mewakili data suhu, label "1" mewakili data PM 1,0, label "2" mewakili data PM 2,5, label "0" mewakili data PM 10, dan label "v" mewakili data kecepatan angin. Bagian kedua dari gabungan data merupakan nilai hasil pembacaan. Nilai-nilai pembacaan pada bagian kedua saling dipisahkan oleh karakter pemisah berupa koma (","). Urutan nilai pembacaan sesuai dengan urutan label pada bagian pertama. Untuk sensor yang tidak mengirimkan data, data sensor tersebut tidak akan digabungkan dengan nilai hasil pembacaan sensor lain. Misalnya, sensor kecepatan angin tidak memberikan nilai pembacaan. Maka, tidak akan ada label "v" pada bagian pertama dan jumlah nilai hasil pembacaan pada bagian kedua juga berkurang satu.

IV. HASIL DAN PEMBAHASAN

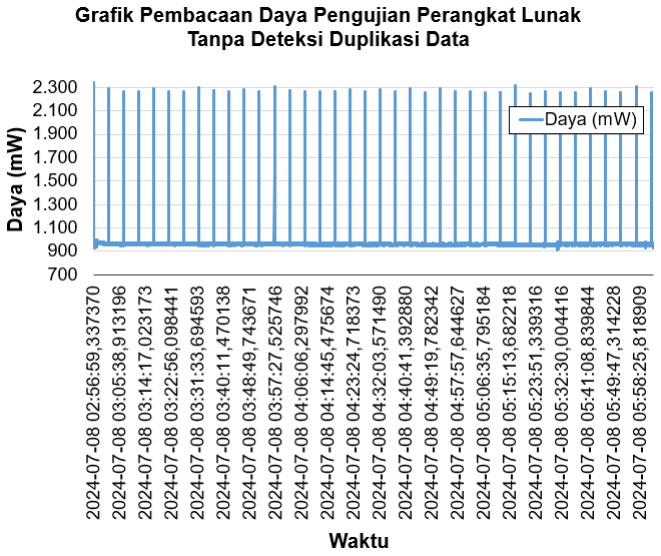
A. PENGUJIAN PERANGKAT LUNAK TANPA DETEKSI DUPLIKASI DATA

Pengujian perangkat lunak tanpa deteksi duplikasi data dilakukan dalam durasi tiga jam, dengan data yang diambil berupa data yang dikirimkan oleh *node* beserta dengan waktu pengirimannya; data yang diterima oleh server beserta dengan waktu penerimaannya; serta data konsumsi daya oleh *node*. Data yang dikirimkan selama pengujian adalah data PM 1,0, PM 2,5, PM 10, suhu, kelembaPan udara, dan kecepatan angin, dengan masing-masing data merupakan nilai acak.

Untuk mengetahui data konsumsi daya pengiriman data, modul LoRa pada sisi *node* dihubungkan secara seri dengan sensor INA219 seperti yang ditunjukkan pada Gambar 3. Hasil pembacaan sensor INA219 didapatkan dari monitor serial Arduino IDE. Sementara itu, Gambar 4 menunjukkan grafik dari konsumsi daya listrik oleh *node* pada saat pengujian dengan durasi tiga jam. Rata-rata konsumsi daya selama tiga jam oleh *node* adalah sebesar 962,93 mW dan untuk setiap pengiriman data diperlukan daya sebesar 2.266,04 mW.

B. PENGUJIAN PERANGKAT LUNAK MENGGUNAKAN DETEKSI DUPLIKASI DATA

Pengujian perangkat lunak dengan deteksi duplikasi data dilakukan mengikuti rangkaian pada Gambar 4. Pengujian ini adalah pengujian program sistem yang telah diperbarui, sehingga terdapat metode deteksi duplikasi data di dalamnya. Pengujian dilakukan untuk mencari akurasi data serta daya listrik yang dibutuhkan saat pengiriman data oleh modul LoRa dengan penggunaan deteksi duplikasi data pada sistem pemantauan kualitas udara. Pengujian dilakukan dalam tiga skema. Skema pertama menggunakan data asli atau data yang selalu sama dalam durasi tiga jam. Skema kedua menggunakan data acak; data hasil pengukuran sensor pada data yang akan dikirim merupakan data acak yang dibuat oleh Arduino Mega. Skema ketiga menggunakan data indeks server yang dikirim oleh server dengan ukuran yang besar. Tiap skema dilakukan



Gambar 4. Grafik pembacaan daya pengujian perangkat lunak tanpa deteksi duplikasi data.

dengan durasi tiga jam, sehingga jumlah percobaan dengan menggabungkan ketiga skema adalah sembilan jam percobaan.

Gambar 5 memperlihatkan hasil pengujian perangkat lunak menggunakan deteksi duplikasi data dengan data pembacaan sensor yang dikirimkan selalu sama pada tiap periode pembacaannya, sehingga data yang dikirimkan selalu sama pada setiap periode pengiriman. Didapatkan hasil bahwa dalam tiga jam durasi pengujian, *node* mengonsumsi daya dengan rata-rata 950,211 mW dan rata-rata daya pada saat pengiriman data sebesar 2.230,51 mW. Nilai tersebut lebih kecil 35,49 mW atau 1,59% lebih kecil daripada pengiriman data tanpa penggunaan metode deteksi duplikasi data.

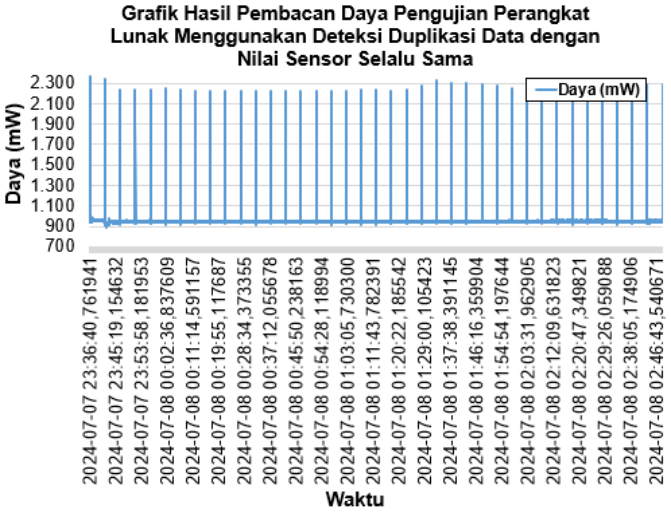
Gambar 6 merupakan hasil pembacaan daya dari pengujian perangkat lunak dengan deteksi duplikasi data dengan hasil pembacaan sensor acak, sehingga data yang dikirim oleh *node* dapat berbeda pada setiap periode pengiriman. Didapatkan hasil bahwa dalam durasi tiga jam pengujian, *node* rata-rata mengonsumsi daya sebesar 937,039 mW, dengan rata-rata daya pada saat pengiriman data sebesar 2.195,026 mW. Nilai tersebut 71,01 mW atau 3,24% lebih kecil dari daya pada pengiriman dalam pengujian perangkat lunak yang tidak menerapkan metode deteksi duplikasi data.

Gambar 7 adalah data konsumsi daya *node* saat pengujian selama tiga jam menggunakan skema data indeks server yang dikirimkan oleh server dengan ukuran besar. Rata-rata konsumsi daya oleh *node* pada saat pengujian berjalan adalah 807,30 mW, dengan rata-rata konsumsi daya saat pengiriman data 2.182,038 mW. Nilai ini 3,85% lebih kecil daripada konsumsi daya tanpa penggunaan deteksi duplikasi data.

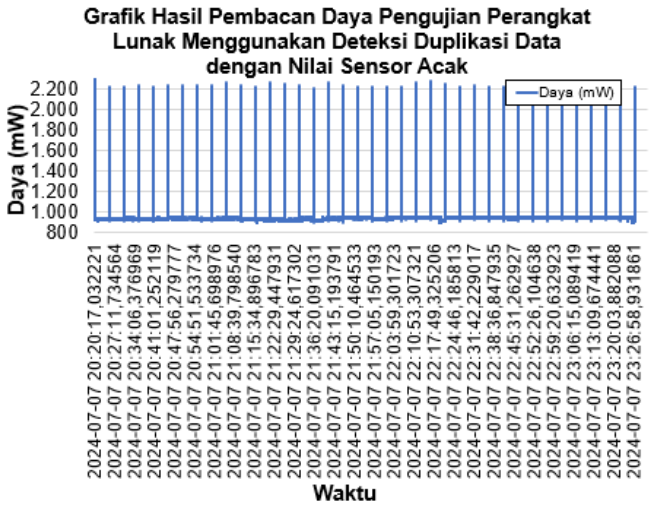
Berdasarkan hasil pengujian pada Gambar 4 sampai Gambar 7, metode deteksi duplikasi data terbukti berhasil menurunkan konsumsi daya. Penurunan ini terjadi karena metode tersebut mengurangi jumlah data yang diterima oleh modul LoRa, yang berdampak pada beban kerja yang lebih ringan. Dengan lebih sedikit data yang diproses, konsumsi daya modul LoRa pun berkurang.

C. PENGUJIAN SISTEM

Pengujian sistem merupakan pengujian untuk mengetahui apakah sistem yang dibangun bekerja dengan baik atau tidak. Hal tersebut dibuktikan dengan tampilnya data hasil pembacaan sensor PM1,0, PM 2,5, PM 10, suhu, kelembapan udara, serta kecepatan angin pada aplikasi android. Pengujian



Gambar 5. Grafik hasil pembacaan daya pengujian perangkat lunak menggunakan deteksi duplikasi data dengan data pembacaan sensor selalu sama.



Gambar 6. Grafik hasil pembacaan daya pengujian perangkat lunak menggunakan deteksi duplikasi data dengan data pembacaan sensor acak.

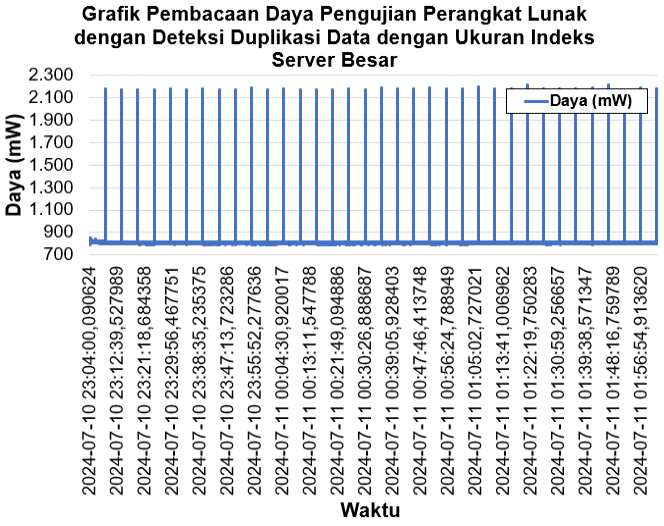
dilakukan dengan membandingkan hasil pembacaan pada *node* dengan hasil pembacaan yang diterima pada aplikasi android dalam durasi 24 jam percobaan. Apabila data yang tampil adalah sama antara keduanya maka dapat disimpulkan bahwa sistem bekerja dengan baik.

Tabel I menampilkan sepuluh data pertama dan sepuluh data terakhir dari hasil pengujian sistem yang dilakukan selama 24 jam. Berdasarkan tabel tersebut, terlihat bahwa hasil pembacaan pada *node* dan data yang diterima oleh aplikasi Android adalah identik. Hal ini menunjukkan bahwa sistem deteksi duplikasi data berfungsi dengan baik, yang terbukti dari kesamaan data antara hasil pembacaan sensor dan data yang ditampilkan pada aplikasi Android.

Pengujian sistem menunjukkan bahwa meskipun terjadi manipulasi data sebelum pengiriman dari *node*, sistem IoT tetap mampu mengirimkan data hasil pembacaan sensor terbaru tanpa kesalahan. Kesalahan yang dimaksud dapat terjadi jika indeks server yang digunakan salah, yang menyebabkan pengiriman indeks server tidak sesuai dengan posisi *data chunk* yang dimaksud di basis data. Kesalahan lainnya dapat muncul jika urutan data pada deteksi duplikasi tidak tepat. Hal ini akan menyebabkan penggabungan *data chunk* yang tidak sesuai, sehingga data yang diterima aplikasi Android menjadi acak. Misalnya, *data chunk* yang seharusnya berada di depan malah

TABEL I
HASIL PENGUJIAN SISTEM

PM 1,0		PM 2,5		PM 10		Suhu		Kelembapan Udara		Kecepatan Angin	
Node	*Android*	*Node*	*Android*	*Node*	*Android*	*Node*	*Android*	*Node*	*Android*	*Node*	*Android*
25	25	34	34	41	41	26	26	77	77	0	0
28	28	36	36	45	45	26	26	77	77	20	20
28	28	36	36	45	45	26	26	78	78	0	0
25	25	34	34	43	43	26	26	78	78	0	0
22	22	30	30	38	38	26	26	79	79	0	0
…	…	…	…	…	…	…	…	…	…	…	…
26	26	34	34	39	39	35	35	55	55	0	0
26	26	34	34	39	39	35	35	55	55	0	0
23	23	31	31	35	35	35	35	55	55	0	0
23	23	31	31	35	35	35	35	56	56	0	0
23	23	32	32	35	35	35	35	54	54	0	0



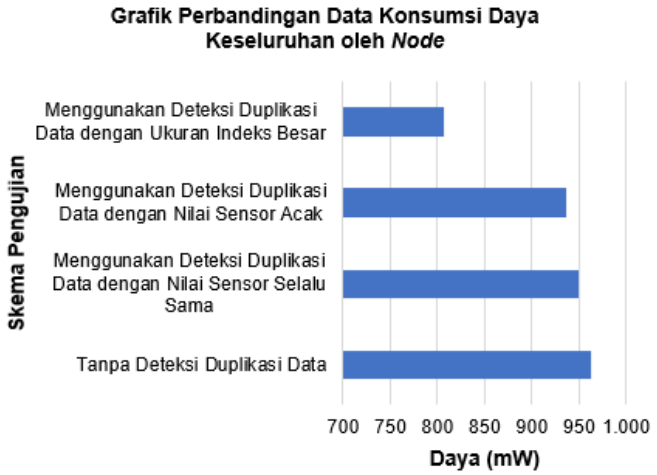
Gambar 7. Grafik hasil pembacaan daya pengujian perangkat lunak menggunakan deteksi duplikasi data dengan data indeks server berukuran besar.

berada di belakang atau *data chunk* yang seharusnya ada di tengah justru ada di depan. Akibatnya, aplikasi Android tidak dapat memproses data tersebut karena struktur data tidak sesuai, sehingga tidak dapat menampilkan data pembacaan sensor.

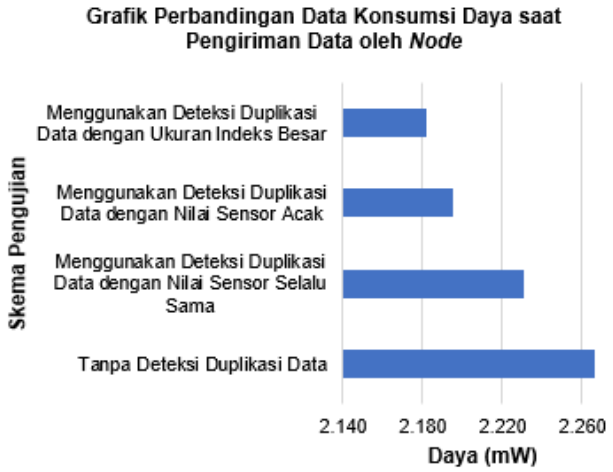
V. ANALISIS

Berdasarkan Gambar 8, terlihat bahwa penggunaan deteksi duplikasi data dengan data indeks server menghasilkan ukuran yang lebih besar, tetapi mencatatkan nilai rata-rata terkecil dibandingkan dengan pengujian lainnya. Hal ini disebabkan oleh konsumsi daya yang lebih rendah saat *node* berada dalam proses tunda, yang hanya mengonsumsi daya sekitar 800 mW, lebih rendah dibandingkan dengan kisaran 900 mW pada pengujian lainnya.

Gambar 9 menunjukkan perbandingan konsumsi daya rata-rata oleh *node* pada setiap pengujian saat pengiriman data. Pengujian dengan deteksi duplikasi data menggunakan skenario ukuran data indeks server besar menunjukkan penurunan konsumsi daya sebesar 3,85% dibandingkan dengan pengujian tanpa deteksi duplikasi data. Pengujian lainnya dengan deteksi duplikasi data pada nilai sensor acak dan nilai sensor selalu sama menunjukkan penurunan konsumsi daya masing-masing sebesar 3,24% dan 1,59%. Hal ini membuktikan bahwa deteksi duplikasi data dapat menurunkan konsumsi daya dengan mengurangi ukuran data yang dikirimkan dari *node* ke *server*, dengan penurunan daya antara 1,59% hingga 3,84%.

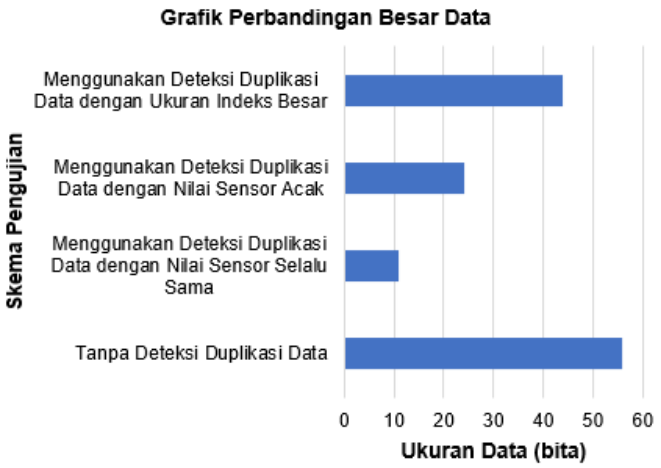


Gambar 8. Perbandingan penggunaan daya keseluruhan untuk sistem tanpa deteksi duplikasi data dan sistem menggunakan deteksi duplikasi data.



Gambar 9. Perbandingan konsumsi daya saat pengiriman untuk sistem tanpa deteksi duplikasi data dan sistem menggunakan deteksi duplikasi data.

Gambar 10 menunjukkan perbandingan ukuran data yang dikirimkan oleh *node* untuk sistem yang menerapkan deteksi duplikasi data dan sistem tanpa penerapan deteksi duplikasi data dalam satuan bita. Tampak bahwa penggunaan deteksi duplikasi data dapat menurunkan ukuran data menjadi berukuran 11-44 bita dari ukuran data tanpa penerapan deteksi duplikasi data sebesar 56 bita. Pada saat data indeks hampir mencapai maksimal, ukuran data indeks adalah 19 bita. Maka, dapat disimpulkan bahwa penggunaan deteksi duplikasi data membuat ukuran data dari 58 bita menjadi 40-45 bita.



Gambar 10. Perbandingan ukuran data yang dikirimkan oleh perangkat lunak tanpa penggunaan deteksi duplikasi data dan perangkat lunak dengan penggunaan deteksi duplikasi data.

Berdasarkan pengujian sistem, diketahui pula bahwa metode deteksi duplikasi data dapat berjalan dengan baik ketika *node* telah terkoneksi dengan sensor-sensor. Hal tersebut dibuktikan pada pengujian sistem, yaitu bahwa nilai dari setiap hasil pembacaan sensor memiliki kesamaan antara nilai pada *node* dengan nilai pada aplikasi Android. Nilai-nilai tersebut berupa nilai pembacaan PM 1,0, PM 2,5, PM 10, suhu, kelembapan udara, dan kecepatan angin.

VI. KESIMPULAN

Sistem deteksi duplikasi data berhasil diterapkan pada sistem pemantauan kualitas udara, terbukti dengan kesamaan nilai sensor PM 1.0, PM 2.5, PM 10, suhu, kelembapan udara, dan kecepatan angin antara aplikasi Android dan *node*. Selain itu, deteksi duplikasi data mampu mengurangi ukuran data yang dikirimkan, dari 56 bita menjadi antara 11 hingga 44 bita. Pengurangan ini terjadi ketika data yang dikirimkan mengandung bagian yang mirip dengan data (*chunk*) sebelumnya. Dalam hal ini, *data chunk* yang sudah ada digantikan dengan indeks dari server, bukan data aslinya, sehingga data yang dikirimkan lebih efisien. Jika data yang terdeteksi merupakan data baru, data tersebut tetap dikirim utuh dan diproses oleh server. Setelah lebih dari sepuluh data unik terdeteksi pada pengiriman pertama, data-data selanjutnya akan selalu dianggap unik meskipun sebenarnya ada data yang terduplikasi. Dalam kondisi ideal tanpa kesalahan komunikasi, penerapan deteksi duplikasi data dapat mengurangi ukuran data hingga 11 hingga 44 bita dan menghemat daya sebesar 1,59% hingga 3,84% dibandingkan dengan sistem tanpa deteksi duplikasi data.

KONTRIBUSI PENULIS

Konseptualisasi, Dwi Ilham Maulana, Agus Purnomo, dan Asep Andang; metodologi, Asep Andang; perangkat lunak, Dwi Ilham Maulana; validasi, Dwi Ilham Maulana, Asep Andang, dan Agus Purnomo; analisis formal, Agus Purnomo; investigasi, Dwi Ilham Maulana dan Asep Andang; sumber daya, Ifkar Usrah; kurasi data, Asep Andang dan Agus Purnomo; penulisan—penyusunan draf asli, Dwi Ilham Maulana; penulisan—peninjauan dan penyuntingan, Dwi Ilham Maulana dan Asep Andang; visualisasi, Dwi Ilham Maulana; pengawasan, Ifkar Usrah; administrasi proyek, Ifkar Usrah; akuisisi pendanaan, Agus Purnomo.

UCAPAN TERIMA KASIH

Terima kasih disampaikan kepada Kementerian Kesehatan Republik Indonesia lewat hibah SIMLITABKES 2024 melalui Poltekkes Kemenkes Tanjung Karang Bandar Lampung yang telah mendanai kegiatan penelitian ini. Terima kasih dihaturkan juga kepada Jurusan Teknik Elektro Universitas Siliwangi yang telah bekerja sama dalam penelitian ini.

REFERENSI

[1] J. Wang dan S. Ogawa, "Effects of meteorological conditions on PM2.5 concentrations in Nagasaki, Japan," *Int. J. Environ. Res. Public Health*, vol. 12, no. 8, hal. 9089–9101, Agu. 2015, doi: 10.3390/ijerph120809089.

[2] R. Kyburz, "Energy efficiency of the Internet of things," 2016. [Online]. Tersedia: https://www.iea-4e.org/wp-content/uploads/publications/2016/08/160704_EE-IoT-Policy-Options_v1.8_-_FINAL_with_cover.pdf

[3] A.S.H. Abdul-Qawy dan T. Srinivasulu, "Greening trends in energy-efficiency of IoT-based heterogeneous wireless nodes," dalam *Int. Conf. Electr. Electron. Comput. Commun. Mech. Comput. (EECCMC)*, 2018, hal. 1–10.

[4] R. Raut dkk., *Green Internet of Things and Machine Learning*. Hoboken, NJ, AS: John Wiley & Sons, 2022.

[5] B. Mahapatra dan A. Nayyar, *Green Internet of Things*. Boca Raton, FL, AS: CRC Press, 2022.

[6] F.K. Shaikh, S. Zeadally, dan E. Exposito, "Enabling technologies for green internet of things," *IEEE Syst. J.*, vol. 11, no. 2, hal. 983–994, Jun. 2017, doi: 10.1109/JSYST.2015.2415194.

[7] J. Botero-Valencia, L. Castano-Londono, D. Marquez-Viloria, dan M. Rico-Garcia, "Data reduction in a low-cost environmental monitoring system based on LoRa for WSN," *IEEE Internet Things J.*, vol. 6, no. 2, hal. 3024–3030, Apr. 2019, doi: 10.1109/JIOT.2018.2878528.

[8] X. Zhang dan M. Deng, "An overview on data deduplication techniques," dalam *Inf. Technol. Intell. Transp. Syst.*, 2016, hal. 359–369, doi: 10.1007/978-3-319-38771-0_35.

[9] X. Liu dan N. Ansari, "Toward green IoT: Energy solutions and key challenges," *IEEE Commun. Mag.*, vol. 57, no. 3, hal. 104–110, Mar. 2019, doi: 10.1109/MCOM.2019.1800175.

[10] "Pengendalian Pencemaran Udara," Peraturan Pemerintah, No. 41, 1999.

[11] "Pengendalian Pencemaran Udara," Peraturan Menteri Lingkungan Hidup dan Kehutanan Republik Indonesia, No 14, 2020.

[12] P. Abbareddy, S. Bhukya, C. Narsingoju, dan B. Narsimhulu, "A novel methodology for secure deduplication of image data in cloud computing using compressive sensing and random pixel exchanging," *J. Theor. Appl. Inf. Technol.*, vol. 102, no. 4, hal. 1608–1618, Feb. 2024.

[13] R. Kaur, I. Chana, dan J. Bhattacharya, "Data deduplication techniques for efficient cloud storage management: A systematic review," *J. Supercomput.*, vol. 74, hal. 2035–2085, Mei 2018, doi: 10.1007/s11227-017-2210-8.

[14] S. Michiels, Ed. Companion, *'08: Proceedings of the ACM/IFIP/USENIX Middleware '08 Conference Companion*. New York, NY, AS: Association for Computing Machinery, 2008.

[15] J. Malhotra dan J. Bakal, "A survey and comparative study of data deduplication techniques," dalam *2015 Int. Conf. Pervasive Comput. (ICPC)*, 2015, hal. 1–5, doi: 10.1109/PERVASIVE.2015.7087116.

[16] J. Riley, *Understanding Metadata*. Baltimore, MD, AS: National Information Standards Organization, 2004.

[17] W. Chen dkk., "Low-overhead inline deduplication for persistent memory," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 8, hal. 1–13, Agu. 2021, doi: 10.1002/ett.4079.

[18] A. Rayes dan S. Salam, "Internet of things (IoT) overview," dalam *Internet of Things from Hype to Reality*. Cham, Swiss: Springer, 2019, hal. 1–35.

[19] R. Arshad dkk., "Green IoT: An investigation on energy saving practices for 2020 and beyond," *IEEE Access*, vol. 5, hal. 15667–15681, Jul. 2017, doi: 10.1109/ACCESS.2017.2686092.

[20] B. Pan dkk., "Study on image encryption method in clinical data exchange," dalam *2015 7th Int. Conf. Inf. Technol. Med. Educ. (ITME)*, 2015, hal. 252–255, doi: 10.1109/ITME.2015.98.