Jurnal Riset Inovasi Bidang Informatika Dan Pendidikan Informatika (KERNEL)
Vol. 4, No. 1, Juli 2023
ISSN: 2774-4345, DOI : 10.31284/j.kernel.2023.v4i1.5514

Komunikasi Multi-Perangkat dan Manajemen Aliran Data dalam Sistem *Internet of Things* Menggunakan Node-Red

Isa Albanna[1], Bagus Kurniawan[2]*

Jurusan Sistem Informasi FTETI Institut Teknologi Adhi Tama Surabaya[1,2]

**Penulis Korespondensi: isaalbanna@itats.ac.id*

ABSTRACT

The Internet of Things can be seen as an integrated system capable of carrying out information exchange, automatic control, and data communication between devices in the internet network. The ESP32 is an embedded system device with additional WiFi architectural features as a communication medium. In this research, a multi-device data communication and data management system was designed using Node-Red. The broker's role as data exchange access functions Node-red as an alternative to the MQTT protocol. The research method carried out includes five stages, namely literacy, design of IoT-ESP32 devices and sensor systems, IoT network configuration, integration of Node-Red and MySQL databases, and testing. According to the test results obtained from the IoT system that was built, data communication between EPS32 devices via the Node-Red intermediary can carry out data transactions well. The Node-Red Broker subscribe and publish process runs appropriately according to the type of information classification required between the system and the user (operator). The response time results in testing data transmission for the monitoring process, data can be sent via the GET method with a response time of 0.00404 seconds (data packet size is 28 bytes and RSSI size is -63 to -60 dB). The response time for reading data by ESP32 to database recording request status conditions is 0.0149 seconds. The database recording request interval time in the research was divided into five variables, and it was found that the sending interval value was greater than 2 seconds. Using a data storage request interval of less than 1 second has an error percentage of around 30–64%.

Article History

Received : 22-12-2023
Revised : 23-12-2023
Accepted : 30-12-2023

Key words

Data Communication
Device ESP32
Flow-based programming
Internet of Things
Node-Red

ABSTRAK

Internet of Things dapat dilihat sebagai sistem terpadu yang mampu menjalankan pertukaran informasi, kendali otomatis dan komunikasi data antar perangkat dalam jaringan internet. ESP32 merupakan perangkat *embedded system* dengan tambahan fitur arsitektur wifi sebagai media komunikasi. Pada penelitian ini dirancang sistem komunikasi data multi perangkat dan manajemen data menggunakan Node-Red. Peran *broker* sebagai akses pertukaran data, memfungsikan Node-red sebagai alternatif dari protokol MQTT. Metode penelitian yang dilakukan meliputi lima tahap yaitu literasi, rancang bangun perangkat IoT-ESP32 dan sistem sensor, konfigurasi jaringan IoT, integrasi Node-Red dan basis data MySQL dan pengujian. Hasil pengujian yang didapatkan dari sitem IoT yang dibangun, komunikasi data antar perangkat EPS32 melalui perantara Node-Red dapat dilakukan transaksi data dengan baik. Proses subscribe dan publish Node-Red *Broker* berjalan dengan tepat sesuai dengan jenis klasifikasi infommasi yang dibutuhkan antar sistem dan pengguna (operator). Hasil respon waktu dalam pengujian pengiriman data untuk proses *monitoring*, data dapat dikirim melalui metode GET dengan waktu respon adalah 0.00404 detik (besar paket data adalah 28byte dan besar RSSI adalah -63 s/d -60dB). Waktu respon untuk pembacaan data oleh ESP32 terhadap kondisi status permintaan perekaman basis data adalah 0.0149 detik. Waktu interval permintaan perekaman basis data dalam penelitian dibuat lima variable dan didapatkan nilai interval pengiriman adalah lebih besar sama dengan 2 detik. Penggunaan interval permintaan penyimpanan data dibawah 1 detik memiliki prosentase errror sekitar 30-64%.

PENDAHULUAN

Peran *Internet of Things* dalam dunia teknologi informasi dapat dipandang sebagai robot yang bekerja secara otomatis dan mampu melayani suatu perintah atau permintaan informasi. Perangkat IoT yang terintegrasi dalam sistem tertanam (*embedded system*) dapat secara cepat, tepat dan efisien untuk melayani berbagai pekerjaan yang menghubungkan berbagai perangkat dalam sebuah jaringan internet. Istilah IoT telah memberikan kontribusi besar dalam pertumbuhan ekonomi, kendali perangkat, akses pertukaran informasi bidang *big-data* dan konsep *machine to machine* (M2M). Perangkat IoT dapat melakukan agregasi data dengan cepat dan konsisten, sehingga dapat mengantikan peran manusia dalam proses *input* informasi dalam sistem penyimpanan basis data,

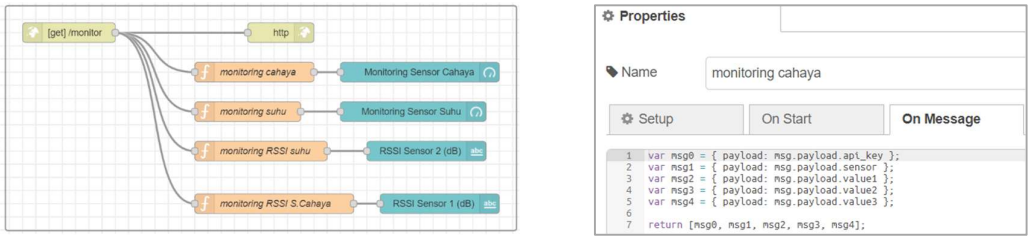
seperti pada implementasi sistem keamanan pada sebuah bangunan [1]. IoT juga mampu secara cepat mengatur pengambilan keputusan melalui kecerdasan buatan yang tertanan dalam logika komputasi. Hal ini dapat dirujuk pada pemanfaatan IoT sebagai kendali ruang penyemaian [2], sistem *monitoring* kualitas udara[3], sistem *smart home* [4] dan *battery management system* (BMS)[5]. Perkembangan IoT selaian dalam arah implementasi, beberapa riset pada bidang komunikasi data juga dilakukan untuk melihat tingkat efektivitas layanan protokol dalam manajemen akses data IoT. Proses pertukaran data dalam sitem IoT sangat cepat dan dilakukan berdasarkan atas permintaan-respon (Request – Respond) antar sistem atau alat yang bersifat otomatis. Penggunaan MQTT (*Message Queue Telemetry Transpor)* dalam manajemen komunikasi IoT banyak digunakan sebagai entitas fungsional layanan transaksi informasi[6].

Protokol MQTT merupakan platform yang mengatur, melayani dan perantara komunikasi data antar alat ke alat lainya. Pada komunikasi MQTT bentuk layanan komunikasi data diataranya meliputi *publisher, broker, dan subscriber*. Metode publisher dalam MQTT adalah pengiriman data oleh perangkat setelah dilakukan permintaan (*subscribe*) oleh alat lain yang bersifat setara sebagai unit klien [7]. Pada studi kasus IoT MQTT juga berperan sebagai perantara (*broker*) yang melayani komunikasi data antar mesin layanan (*server*) dan klien (*client*)[7]. Pertanyaan besar dalam pengembangan sistem IoT terhadap keberadaan protokol MQTT adalah apakah MQTT bertindak sebagai server atau hanya pihak ketiga yang berbeperan sebagai manajamen data. Apabila keberadaan MQTT merupakan layanan pihak ketiga, apakah dapat dicari alternatif sistem manajemen data IoT yang *endurance*, efisien dan efektiv. Hal tersebut mendasari peneliti pengembangkan Node-Red sebagai manajemen komunikasi, data dan komputasi dalam IoT. Node-Red merupakan platform komputasi berbasis daring dan memiliki pola komputasi bersifat aliran data atau *Flow-based programming* (FBP)[8]. Node-Red dikembangkan oleh IBM dan OpenJS sebagai pemrograman berbasis layanan (server) dalam jaringan internet. Tujuan penelitian ini adalah bagaimana memfungsikan Node-Red sebagai platform yang mampu mengatur komunikasi antar perangkat IoT, Node-Red mampu memberikan layanan (Req-Res) dalam akses cepat pertukaran data, dan Node-Red sebagai komputasi pendukung dalam IoT sehingga dapat alternatif layanan MQTT.

TINJAUAN PUSTAKA

Node-Red

Pada tahun 2013 Node-Red merupakan *platform* yang dikembangkan oleh Nick-O'Leary dan Dave Conway-Jones dari Tim IBM UK's *Emerging Technology Services* dan pada tahun 2016 Node-Red telah diakusisi oleh Open JS Foundation [9]. Node-Red merupakan bahasa pemrograman *hybrid* yang memadukan blok fungsional yang nantinya dapat terhubung dalam node-flow dan masing-masing blok fungsional dalam dilakukan pengaturan dengan menggunakan bahasa pemrograman JavaScript. Node-Red membutuhkan *runtime* node.js dalam menjalankan seluruh aktivitas. Node-Red bersama node.js menjadi sebuah *server* khusus yang berjalan pada port default yaitu 1880 untuk melayani permintaan dari pengguna. Pemrograman Node-Red bersifat *Flow-based programming* (FBP) yang artinya seluruh alur pemrograman didesain dalam aliran blok yang terangkai untuk membentuk perintah kerja. Blok-blok fungsional disusun sedemikian rupa untuk menangani pekerjaan yang digunakan dalam komunikasi data atau *dashboard* IoT. Pada Gambar 1 ditunjukkan contoh studi kasus blok fungsional Node-Red sebagai pemilan data objek dan penggunaan pemrograman JavaScript dalam blok fungsi. Modular blok yang tersedia dalam Node-Red mendukung secara penuh perancangan sistem IoT [8]. Terdapat beberapa varian modul yang dapat dimodifikasi untuk membentuk sistem IoT, diantaranya dalah modul komunikasi *socket* HTTP, modul komunikasi serial-COM, masukan data-file, kendali, modul basis data dan modul UI-*Dashboard*. Modul-modul yang ada dalam Node-Red membentuk kesatuan yang nantinya terhubung dalam diagram alir. Proses aliran data dalam Node-Red dibagi menjadi data *context* yaitu *node*, *flow* dan *global*. Masing-masing tipe data tersebut pada penelitian digunakan sebagai referensi antar aliran diagram. Variabel dalam masing-masing fungsi didesain dengan merelasikan data *context* yang masih terdapat dalam satu proyek desain aliran.



(a) (b)

Gambar 1. a) Diagram alir untuk penyusunan program *block* dalam Node-Red, b) Pemrograman JavaScript untuk pengambilan data obyek dalam *block-function* Node-Red [8].

Perangkat ESP32 dan Sistem Sensor

ESP32 merupakan *System-on-a-Chip* (SoC) yang diproduksi oleh perusahaan Espressif. Modul ESP32 selain sebagai SoC yang mampu mengendalikan perangkat, modul tersebut juga memiliki bagian komunikasi wifi yang dapat terhubung dalam sebuah jaringan. Arsitektur ESP32 terbagi menjadi dua blok yaitu blok sistem tertanam (*Embedded System*) dan blok *Network Radio Frequency* [10]. Pada bagian *embedded system*, SoC tersebut didukung dengan keberadaan GPIO yang dapat berkomunikasi dengan perangkat lain melalui sinyal data elektronik. Protokol komunikasi *micro-electronic* yang dapat diakomodir oleh ESP32 meliputi komunikasi I2C, Serial Port-COM RS232, I/O, *one-wire*, TWI (*two wire interface*) dan SPI (*Serial Peripheral Interface*)[11]. Beberapa bentuk komunikasi tersebut yang nantinya dapat mendukung dari antar-muka ESP32 dan kedua modul sensor dalam penelitian (modul sensor Cahaya BH1750 dan modul sensor suhu DS18B20). Pada Gambar 2a, merupakan morfologi fisik dari ESP32 dengan dalam papan esp32 devkit v1. Selain memiliki perangkat GPIO sebagai penghubung antar perangkat elektronik, ESP32 memiliki blok RF yang mencakup komunikasi berbasis Wifi, Bluetooth dan *Bluetooth Low Energy* (BLE). Dalam penelitian modul RF yang digunakan hanya difokuskan pada modul Wifi yang nantinya akan digunakan dalam komunikasi TCP/IP. Pada Gambar 2b merupakan morfologi fisik dari sensor Cahaya. Sensor Cahaya dengan jenis BH1750 merupakan sensor yang diproduksi oleh ROHM *semiconductor* dengan range kerja pengukuran intensitas cahaya adalah 1 – 65535 lux [12]. Komunikasi sensor BH1750 dengan ESP32 adalah menggunakan jalur protokol I2C. Gambar 2c merupakan bentuk fisik sensor suhu DS18B20 yang diproduksi oleh *maxim integrated* dengan besar pengukuran suhu adalah -55°C sampai dengan 125°C[13].



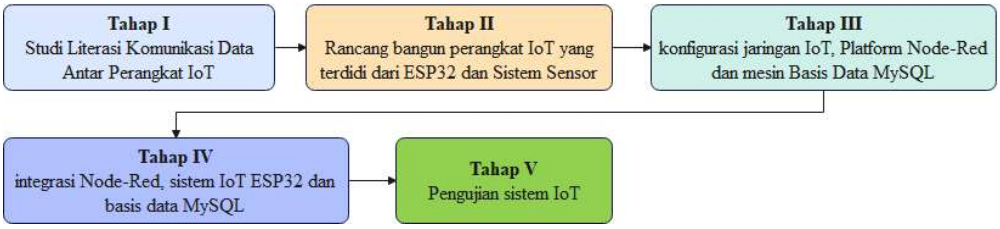
(a) (b) (c)

Gambar 2. a) Modul ESP32 dengan GPIO, b) Sensor cahaya BH1750, c) Sensor suhu DS18B20.[12], [13], [14]

METODE

Pada penelitian terkait komunikasi multi-perangkat dan manajemen aliran data dalam sistem *internet of things* menggunakan Node-Red, terbagi menjadi lima tahap sesuai dengan diagram alir pada Gambar 3, penelitian diawali dengan studi literasi terkait komunikasi data antar perakat dalam jaringan IoT. Kemudian pada tahap kedua dilakukan rancang bangun perangkat keras ESP32 dan modul sistem sensor guna mendukung sistem IoT. Pada tahap ketiga adalah konfigurasi jaringan IoT,

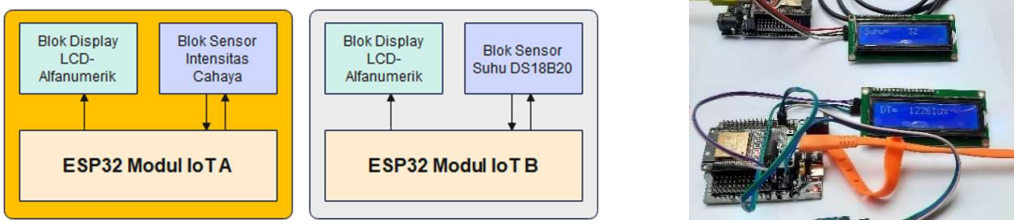
platform server Node-Red dan mesin basis data MySQL. Tahap ke empat adalah integrasi Node-Red, sistem IoT ESP32 dan basis data MySQL. Tahap terakhir (ke lima) adalah pengujian sistem.



Gambar 3. Skema alur penelitian komunikasi multi-perangkat IoT dan manajemen aliran data berbasis Node-Red.

Rancang Bangun Perangkat IoT

Perancangan perangkat keras IoT terdiri dari empat modul yaitu modul papan ESP32 dari WROOM, modul sensor BH1750, sensor suhu DS18B20 dan panel tampilan (*display*) LCD alfanumemik 1602. Seluruh bagian perangakat keras disusun dengan arsitektur seperti pada Gambar 4 diagram susunan perangkat keras IoT. ESP32 pada penelitian ini bertindak sebagai unit agregasi data sensor yang diperankan melalui komunikasi I2C (BH1750) dan *One-Wire* (DS18B20). Satu set modul IoT terdiri dari ESP32, sensor, dan perangkat tampilan LCD. Pada penelitian digunakan dua set modul IoT, dengan konfigurasi pembeda adalah jenis sensor. Sensor pada set IoT A memiliki sensor Cahaya BH1750 sebagai unit pembaca intensitas cahaya dalam satuan lux. Teknik komunikasi digital antara sensor cahaya dan ESP32 menggunakan komunikasi I2C. Pada komunikasi tersebut memungkinkan meminimalisir penggunaan instalasi kabel dan jalur komunikasi berada dalam pin SDA-SCL. Pada set IoT B terpasang sensor DS18B20 satu unit sebagai *monitoring* suhu lingkungan.



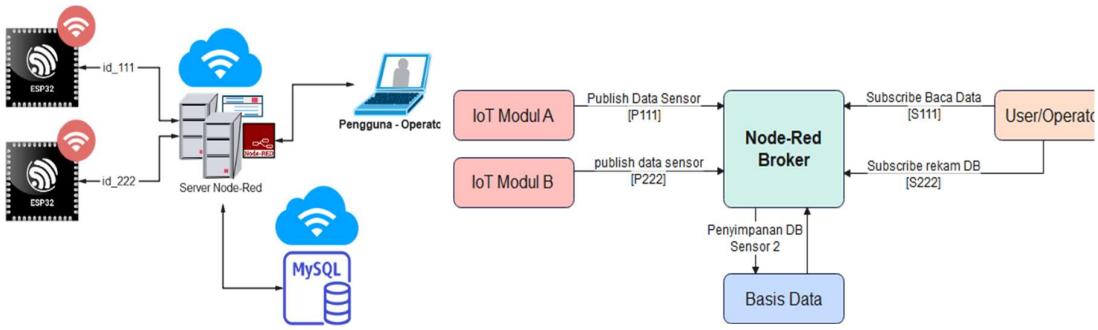
(a) (b)

Gambar 4. a) Desain perangkat keras modul IoT, b) realisasi instrumen IoT yang terdiri dari ESP32, sistem sensor dan perangkat tampilan LCD-Alfanumerik 1602.

Konfigurasi Jaringan dan Platform Node-Red

Sistem IoT yang dibangun dalam penelitian ini, terdiri dari *client-server*. *Client* pada sistem IoT terdiri dari dua skema yaitu modul ESP32 dan pengguna manusia. ESP32 difungsikan sebagai klien yang mampu mengirim data sekaligus membaca permintaan dari klien lainya. Modul ESP32 terprogram secara otomatis untuk melayani permintaan yang dikirim oleh klien-manusia (operator) saat mengoperasikan dashboard UI website IoT. ESP32 dalam studi kasus ini didesain dengan dua fungsional layaknya multi-klien dalam protokol MQTT. ESP32 akan memproses data sensor hingga menyimpan dalam variable sementara pada memori internal EEPROM. Ketika terjadi permintaan dari klien lain atau dalam arti proses *subscribe*, maka sistem ESP32 akan melakukan *publish* data menuju Node-Red. Ilustrasi pertukaran data *Subscribe* dan *Publish* dari modul klien IoT seperti pada Gambar 5, dalam gambar tersebut data akan diberikan *tokenizing* berupa pengalamatan (id_alat)

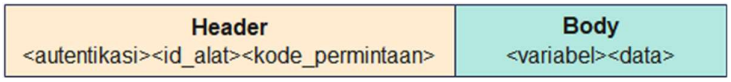
perangkat dalam setiap transmisi data. Hal tersebut agar tidak tertukar dalam penerjemahan data oleh perangkat ESP32.



(a) (b)

Gambar 5. a) Komunikasi antara perangkat dan *server* node-red dengan alamat id_alat, b) konsep pertukaran data dalam IoT untuk *publish* dan *subscribe*.

Teknik komunikasi antara server dan klien digunakan komunikasi pada protokol HTTP melalui jaringan internet. Masing-masing perangkat IoT akan mendapatkan alamat IP yang diberikan oleh *router*-AP. Perangkat ESP32 selalu dalam kondisi terjaga (*standby*) dan akan memberikan tanggapan (respon) ketika terdapat permintaan dalam jalur HTTP. Seluruh transaksi data dari *server* ke klien dilewatkan dapa protokol HTTP dengan struktur format seperti pada gambar 6, dalam komunikasi terdapat dua struktur yaitu *header* dan *body*. Frame *header* berisikan infrormasi autentikasi, jenis data dan alamat dari perangkat. Pada frame tubuh terdapat informasi berupa data yang nantinya dikirim dalam format *msg.payload* yang dapat diterjemahkan oleh perangkat IoT.



Gambar 6. Frame data dalam komunikasi IoT *Client-Server* Node-Red

Integrasi Node-Red dan Basis Data MySQL

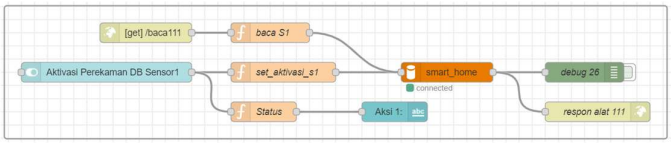
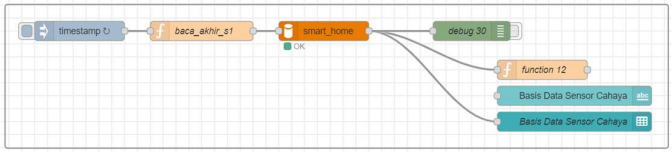
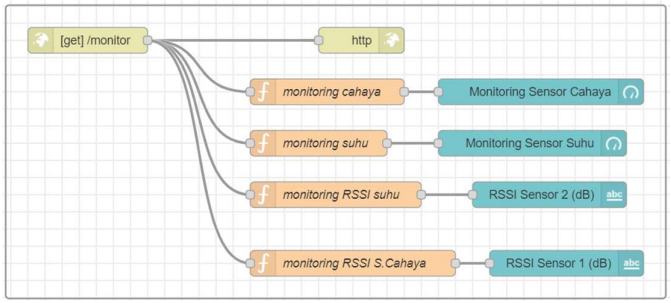
Pada penelitian digunakan Node-Red versi 3.0 dan NodeJS versi 18.0 yang memberikan dukungan dalam manajemen pemrograman berbasis diagram alir (BFP). Pada penelitian dirancang UI-*Dashboard* yang nantinya digunakan untuk mengatur aktivasi perekaman data oleh operator terhadap sistem monitoring sensor. Bentuk rancang bangun UI ditunjukkan seperti pada Gambar 7, dalam gambar tersebut terdapat beberapa panel yang terdiri dari *Gauge monitoring*, tombol geser (*switch*) untuk aktivasi perekaman data dalam basis data MySQL, status *monitoring* RSSI kekuatan sinyal wifi dan panel tabel untuk mengetahui hasil perekaman dalam basis data. Pada sistem tersebut, ESP32 akan terus menerus mengirimkan data pada *dashboard* akan tetapi setiap data yang dikirim tidak dilakukan penyimpanan pada basis data. Ketika pengguna melakukan aktivasi tombol perekaman maka akses informasi akan dialirkan pada mesin basis data MySQL.



Gambar 7. Rancangan GUI untuk klien-operator dalam manajemen perekaman basis data sensor pada perangkat IoT.

Untuk menyusun tampilan dan aktivitas *back-end,* pada platform Node-Red dibentuk beberapa *node* (jaringan antara blok fungsi) untuk mengatur akses komputasi yang mencakup: permintaan-respon (*Req-Res*) data dari protokol HTTP, desain UI-Dashboard interaksi aktivasi oleh operator, dan perintah perekaman data. Pada tabel 1 disajikan seluruh diagram alir Node-Red untuk menjalankan aktivitas sistem IoT.

Tabel 1. Logika Node dalam platform Node-Red

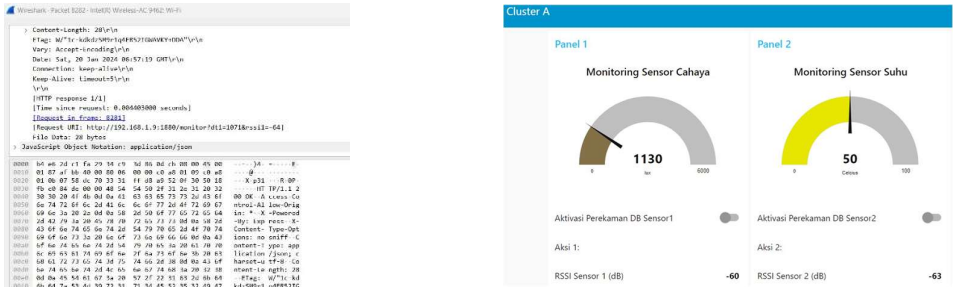
Bentuk Diagram Node	Keterangan aksi dalam node
	Node untuk pembacaan data dari protokol HTTP; aktivasi status perekaman data dan pengiriman respon pada protokol HTTP
	Pemutahiran data pada tampilan tabel yang dirujuk dari basis data MySQL
	Blok fungsi untuk pemisahan data dalam objek yang dikirim dalam protokol HTTP untuk ditampilkan data pada panel monitoring (*gauge* dan *text*)

HASIL DAN PEMBAHASAN

Pengiriman Data Monitoring Sensor

Data hasil pembacaan sensor dari masing-masing perangkat ESP32 akan dikirim pada server Node-Red yang nantinya akan ditampilkan dalam panel monitoring berupa *gauge meter*. Hasil observasi melalui perangkat wireshark didapatkan paket pengiriman data seperti pada Gambar 8 dengan informasi pengiriman melalui perintah HTTP-Request GET. Format permintaan GET (http://alamatdomain/monitor?dt1=<data1>&rssi1=<data_rssi>) terprogram dalam *embedded system* ESP32 dan dikirim secara berulang per satu detik. Dari hasil observasi wireshark didapatkan waktu

respon server Node-Red adalah sekitar 0.004403 detik dengan pengiriman data dalam satu paket adalah 28 byte. Data terkirim pada saat kondisi RSSI dengan besar sekitar -60 sampai dengan -63dB.

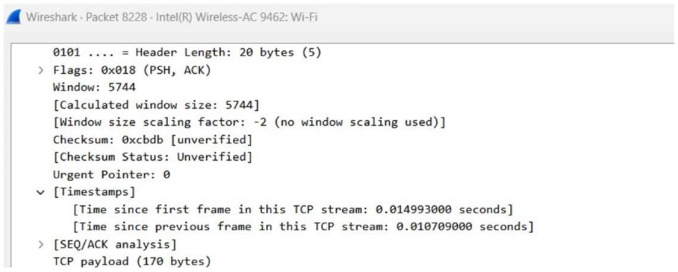


(a) (b)

Gambar 8. a) Waktu respon server Node-Red untuk permintaan monitoring data dari ESP32, b) tampilan RSSI pada panel UI-*Dashboard*.

Pembacaan Data ESP32 Terhadap Permintaan Klien

Kesetaraan posisi ESP32 dan operator sebagai klien dalam sistem IoT membuat ESP32 tidak bisa menerima secara langsung data yang dikirim oleh klien lain. Proses permintaan klien kepada ESP32 dilewatkan melalui *update* basis data MySQL yang secara kontinu dibaca oleh perangkat ESP32. Untuk menjembatani komunikasi data permintaan klien, dalam basis data disediakan parameter "status_alat". Melalui parameter status_alat tersebut, ketika user melakukan *click* aktivasi melalui tombol (*switch*) maka perintah (*query*) update akan tercipta dan akan mengantikan kondisi pada enumerisasi "rekam_stop" dan "rekam_go". Pada saat permintaan rekam stop, maka ESP32 akan mengirimkan permintaan POST dengan format <data header><data> menuju server Node-Red. Sistem Node-Red akan mengolah data dan mendistribusikan pada mesin basis data MySQL. Pada Gambar 9.a merupakan waktu respon *server* dalam menangani permintaan POST. Pada Gambar 9.b merupakan bentuk respon data yang diberikan oleh perangkat ESP32 dari hasil pemantauan komunikasi Serial COM RS232. Dari data pemantauan wireshark didapatkan waktu respon kurang dari 1 detik (0.0149 Detik) untuk pengambilan hingga perekaan data dalam DB-MySQL



(a)



(b)

Gambar 9. a) Waktu respon server untuk perintah POST-Perekaman DB, b) Perekaman waktu pengiriman oleh ESP32 melalui komunikasi seria.

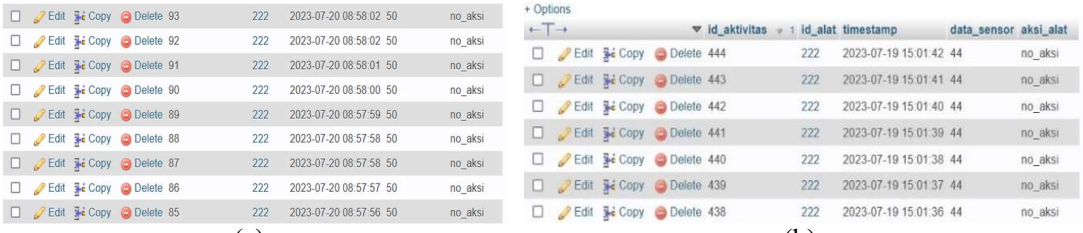
Perekama Data MySQL

Pada penelitian digunakan variasi jeda waktu pengiriman data dari ESP32 menuju Node-Red dan MySQL. Interval jeda waktu yang diberikan adalah sebanyak enam variasi yaitu 5,4,3,2,1 dan 0.5 detik. Pengujian dilakukan pada *bandwidth* internet yang sama (unduh=2.44Mbps dan unggah=1.89Mbps). Hasil pengukuran error data disajikan dalam tabel 2, dari pengujian tersebut terlihat error data yang diindikasikan adanya redudansi data dalam pengukuran dalam satuan waktu. Data uji diambil 50 data pertama dari tiap perubahan waktu jeda/internval.

Tabel 2. Prosentase data error terhadap jeda waktu permintaan perekaman data

No	Jeda waktu *request* POST-Node Red-MySQL	Frekuensi data error	Banyak data	Prosentase Error
1	5	0	50	0
2	4	0	50	0
3	3	1	50	2
4	2	1	50	2
5	1	15	50	30
6	0.5	32	50	64

Hasil observasi pengujian pada interval lebih kecil dari 2 detik didapatkan data error pengiriman yang cukup besar. Pengiriman data dalam interval waktu 1 detik akan memperbesar error yang ditunjukkan seperti pada Gambar 10, dalam perekaman basis data ditunjukkan adanya pengukuran redudansi data pada tiap detiknya. Pengiriman data per 0.5 detik tidak bisa terbaca oleh MySQL yang terjadi perulangan dalam interval waktu yang sama dan terjadi waktu interval pengiriman yang acak dari tiap baris perekaman.



(a) (b)

Gambar 10. a) Pengambilan data perekaman MySQL dengan jeda waktu 0.5 detik, b) pengambilan data perekaman MySQL dengan jeda waktu 1 detik.

***Subscribe* dan *Publish* Perantara Node-Red**

Kinerja Node-Red sebagai perantara informasi atau *broker*, dilakukan dengan skema pengujian permintaan (*Subscribe*) dari operator dengan meng-click *switch* pada *dashboard* dan Node-Red akan memicu untuk ESP32 melakukan pengiriman perintah perekaman data. Perintah perekaman data kemudian diteruskan oleh Node-Red pada mesin server basis data MySQL. Adapun perintah *subcribe* dan *publish* dari sistem ditunjukkan dalam tabel 3, yang memuat aksi dari dampak subscribe dan publish dalam sistem IoT.

Tabel 3. Perintah subscribe-publish oleh *broker* Node-Red

Perangkat	Perintah Subscribe	Node-Red	Respon Publish
Modul 1- ESP32	[GET]/read111	Msg.payload data sensor modul 1	Gauge monitoring sensor 1
Modul 2- ESP32	[GET]/read222	Msg.payload data sensor modul 3	Gauge monitoring sensor 2

Perangkat	Perintah Subscribe	Node-Red	Respon Publish
Klien-Operator	[GET]/rekam	Msg.payload data perekaman ke sensor 1 atau sensor 2	[POST]/rekam <parameter body>
Basis Data	-	Msg.payload dengan topic perintah query "insert"	[POST]/rekam <parameter body sensor 1 atau 2>

KESIMPULAN

Dari serangkaian penelitian yang telah dilaksanakan terkait komunikasi multi-perangkat dan manajemen aliran data dalam sistem *internet of things* menggunakan Node-Red dapat diambil kesimpulan diantaranya: perintah *subscribe* dan *publish* dapat dikendalikan dengan akurat oleh Node-Red yang bertindak sebagai *broker*; pengiriman data *monitoring* sensor dapat dilakukan dengan baik antara ESP32 dan Node-Red dengan waktu respon 0.00404 detik; permintaan data antar klien dapat ditangani oleh Node-Red sebagai perantara (*broker*) dengan eksekusi waktu respon adalah 0.0149 detik; dan waktu jeda (interval) perekaman data antara ESP32 dengan MySQL dengan waktu kurang dari satu detik akan menghasilkan error yang cukup besar yaitu 64%.

DAFTAR PUSTAKA

[1] W. Triyoga, Y. A. Suryo, and R. P. Astutik, "Rancang Sistem Keamanan Pada Laboratorium Berbasis Internet Of Things Menggunakan Rcwl Sebagai Pendeteksi Gerakan," *J. Compr. Sci. JCS*, vol. 2, no. 6, pp. 1593–1606, Jun. 2023, doi: 10.59188/jcs.v2i6.381.

[2] R. Doni and M. Rahman, "Sistem Monitoring Tanaman Hidroponik Berbasis Iot (Internet of Thing) Menggunakan Nodemcu ESP8266," *J-SAKTI J. Sains Komput. Dan Inform.*, vol. 4, no. 2, Art. no. 2, Sep. 2020, doi: 10.30645/j-sakti.v4i2.243.

[3] D. Wall, P. McCullagh, I. Cleland, and R. Bond, "Development of an Internet of Things solution to monitor and analyse indoor air quality," *Internet Things*, vol. 14, p. 100392, Jun. 2021, doi: 10.1016/j.iot.2021.100392.

[4] A. Nuradi, "Pengembangan Arsitektur Otomatisasi Smart Home dengan Internet of Things," *J. Sist. Cerdas*, vol. 1, no. 2, Art. no. 2, Dec. 2018, doi: 10.37396/jsc.v1i2.13.

[5] M. A. Salam and W. Aribowo, "Monitoring dan Kendali Charger Accu Berbasis Node-RED," *J. Tek. ELEKTRO*, vol. 13, no. 1, pp. 14–19, 2024, doi: 10.26740/jte.v13n1.p14-19.

[6] H. Hairatunnisa, H. A. Nugroho, and R. Margiono, "Analisis Kinerja Protokol MQTT dan HTTP Pada Akuisisi Data Magnet Berbasis Internet of Things," *J. Ilm. Inform.*, vol. 6, no. 2, Art. no. 2, Dec. 2021, doi: 10.35316/jimi.v6i2.1351.

[7] E. Longo and A. E. C. Redondi, "Design and implementation of an advanced MQTT broker for distributed pub/sub scenarios," *Comput. Netw.*, vol. 224, p. 109601, Apr. 2023, doi: 10.1016/j.comnet.2023.109601.

[8] T. Hagino, *Practical Node-RED Programming: Learn powerful visual programming techniques and best practices for the web and IoT*. Birmingham: Packt Publishing, 2021.

[9] R. K. Kodali and A. Anjum, "IoT Based Home Automation Using Node-RED," in *2018 Second International Conference on Green Computing and Internet of Things (ICGCIoT)*, Aug. 2018, pp. 386–390. doi: 10.1109/ICGCIoT.2018.8753085.

[10] N. Cameron, *ESP32 formats and communication: application of communication protocols with ESP32 microcontroller*. Berkeley, CA: Apress, 2023.

[11] S. Aheleroff *et al.*, "IoT-enabled smart appliances under industry 4.0: A case study," *Adv. Eng. Inform.*, vol. 43, p. 101043, Jan. 2020, doi: 10.1016/j.aei.2020.101043.

[12] A. Khuriati, "Sistem Pemantau Intensitas Cahaya Ambien Dengan Sensor BH1750 Berbasis Mikrokontroler Arduino Nano," *Berk. Fis.*, vol. 25, no. 3, pp. 105–110, Dec. 2022.

[13] A. Elyounsi and A. N. Kalashnikov, "Evaluating Suitability of a DS18B20 Temperature Sensor for Use in an Accurate Air Temperature Distribution Measurement Network," *Eng. Proc.*, vol. 10, no. 1, Art. no. 1, 2021, doi: 10.3390/ecsa-8-11277.

[14] V. O. Oner, *Developing IoT projects with ESP32: automate your home or business with inexpensive Wi-Fi devices*, First published. Birmingham Mumbai: Packt, 2021.