MODEL JARINGAN MESH PADA SISTEM *INTERNET OF THINGS* (IOT) UNTUK PEMANTAUAN LINGKUNGAN

SKRIPSI

Diajukan sebagai salah satu syarat untuk menyelesaikan
pendidikan diploma empat (D-4) Program Studi Teknik Komputer dan Jaringan
Jurusan Teknik Elektro
Politeknik Negeri Ujung Pandang

AISYAH AMINI A'MAARI SUHUD
425 18 054

PROGRAM STUDI D-4 TEKNIK KOMPUTER DAN JARINGAN
JURUSAN TEKNIK ELEKTRO
POLITEKNIK NEGERI UJUNG PANDANG
MAKASSAR
2022

HALAMAN PENGESAHAN

Skripsi dengan judul "**MODEL JARINGAN MESH PADA SISTEM *INTERNET OF THINGS* (IOT) UNTUK PEMANTAUAN LINGKUNGAN**" oleh **Aisyah Amini A'maari Suhud (425 18 054)** telah diterima dan disahkan sebagai salah satu syarat untuk memperoleh gelar Diploma 4 (D4/S1 Terapan) pada Program Studi Teknik Komputer dan Jaringan Jurusan Teknik Elektro Politeknik Negeri Ujung Pandang

Makassar, 29 Desember 2022

Pembimbing I	Pembimbing II
Drs. Kasim M.T	**Irmawati S.T M.T**
NIP. 196306201991031002	**NIP. 197811242012122002**

Mengetahui,

Koordinator Program Studi
Teknik Komputer dan Jaringan
Politeknik Negeri Ujung Pandang

Eddy Tungadi, S.T., M.T.
Nip.19790823 201012 1 001

HALAMAN PENERIMAAN

Pada hari ini, hari Kamis tanggal 29 Desember 2022 Tim Penguji Sidang Skripsi telah menerima dengan baik hasil skripsi oleh mahasiswa: **Aisyah Amini A'maari Suhud** nomor induk mahasiswa **42518054** dengan judul **MODEL JARINGAN MESH PADA SISTEM *INTERNET OF THINGS* (IOT) UNTUK PEMANTAUAN LINGKUNGAN**

Makassar, 29 Desember 2022

Tim Penguji Ujian Sidang Skripsi

1. Ir. Dahlia, M.T	Ketua	(.....)
2. Iin Karmila Yusri S.ST, M.Eng., Ph.D	Sekretaris	(.....)
3. Rini Nur S.T., M.T.	Anggota	(.....)
4. Zawiyah Saharuna S.T., M.Eng	Anggota	(.....)
5. Drs. Kasim, M.T	Anggota	(.....)
6. Irmawati, S.T., M.T	Anggota	(.....)

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Allah Swt., karena atas segala berkah dan karunia-Nya yang tak henti memberikan kekuatan, kesehatan dan keselamatan sehingga penulis dapat menyelesaikan penyusunan skripsi dengan baik

Skripsi ini disusun untuk memenuhi salah satu persyaratan untuk menyelesaikan studi pada Pogram Studi D-IV Teknik Komputer dan Jaringan di Politeknik Negeri Ujung Pandang.

Penulis menyadari bahwa keberhasilan penyusunan skripsi ini tidak lepas dari bantuan berbagai pihak baik secara langsung maupun tidak langsung. Oleh karena itu, dengan rendah hati penulis mengucapkan terima kasih kepada :

1. Orangtua penulis yakni Ayahanda Much Suhudi dan Ibunda Darnawati serta Adik Kandung Penulis, Muthia Amelia, Much Said Aqil, Much Arwanda yang senantiasa memberikan semangat, motivasi, dukungan, bimbingan dan doa restu kepada penulis.
2. Bapak Prof. Ir. Muhammad Anshar, M.Si.,Ph.D. selaku Direktur Politeknik Negeri Ujung Pandang.
3. Bapak Ahmad Rizal Sultan, S.T.,M.T.,Ph.D. selaku Ketua Jurusan Teknik Elektro Politeknik Negeri Ujung Pandang.
4. Bapak Eddy Tungadi, S.T., M.T. selaku Ketua Program Studi Teknik Komputer dan Jaringan.

5. Bapak Drs. Kasim, M.T. selaku pembimbing I dan Ibu Irmawati S.T., M.T. selaku pembimbing II atas segala ilmu, motivasi, nasehat, arahan, bantuan dan kesedian waktu dan kesabarannya dalam membimbing penulis hingga terselesaikannya penelitian ini.
6. Seluruh dosen dan Staf Jurusan Teknik Elektro, Khususnya Prodi Teknik Komputer dan Jaringan.
7. Teman-teman, Siti Annisa Saharanti, Asrul Said, Hendra, Ahmad Shabri Azis, Evy Wulandari, Darmi. yang membantu penulis dalam segala hal
8. Teman-teman TKJ'18 yang senantiasa memberikan semangat, motivasi dan dukungan serta setia menemani penulis dalam penyusunan laporan skripsi.
9. Semua pihak yang membantu penulis baik secara langsung maupun tidak langsung yang tidak dapat disebutkan satu per satu.

Penulis sadar bahwa dalam penyusunan laporan skripsi ini masih terdapat banyak kekurangan. Untuk itu penyusun mohon maaf dan mengharapkan kritik membangun serta saran terhadap laporan skripsi ini. Semoga apa yang penulis lakukan dapat bermanfaat bagi pembaca agar dapat menghasilkan sumber daya manusia yang berkualitas dan berguna bagi bangsa dan negara.

Makassar, 20 September 2022

Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

DAFTAR TABEL

SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama : Aisyah Amini A'maari Suhud

NIM : 42518054

Menyatakan dengan sebenar-benarnya bahwa segala pernyataan dalam skripsi ini yang berjudul **MODEL JARINGAN MESH PADA SISTEM INTERNET OF THINGS (IOT) UNTUK PEMANTAUAN LINGKUNGAN** merupakan gagasan dan hasil karya saya sendiri dengan arahan komisi pembimbing, dan belum pernah diajukan dalam bentuk apapun pada perguruan tinggi dan instansi manapun.

Semua data dan informasi yang digunakan telah dinyatakan secara jelas dan dapat diperiksa kebenarannya. Sumber informasi yang berasal atau daikutip dari karya yang diterbitkan dari penulis lain telah disebutkan dalam naskah dan dicantumkan dalam skripsi ini.

Jika pernyataan saya tersebut di atas tidak benar, saya siap menanggung resiko yang ditetapkan oleh Politeknik Negeri Ujung Pandang.

Makassar, 29 Desember 2022

Aisyah Amini A'maari Suhud
42518054

Model Jaringan *Mesh* Pada Sistem Internet Of Things (Iot) Untuk Pemantauan Lingkungan

RINGKASAN

Dominasi jumlah penduduk di Indonesia meningkat setiap tahunnya. Angka tersebut meningkat pada 2025 hingga mencapai 170,4 juta jiwa atau 59,3% dari total penduduk Indonesia yang sebesar 287 juta jiwa. Kementrian Lingkungan Hidup dan Kehutanan (KLHK) Memiliki sistem pengukuran kualitas lingkungan di beberapa kota namun pengukuran yang dilakukan oleh KLHK hanya pada satu titik untuk satu kota, sehingga hal ini tidak bisa menjadi acuan kualitas lingkungan untuk seluruh lokasi yang ada. Arsitektur IoT dapat menghubungkan semua perangkat komunikasi dan baik digunakan pengembangannya di berbagai bidang karena sifatnya yang dinamis membawa dampak yang signifikan pada aplikasi berbasis IoT terutama kemampuan beradaptasi dan jangkauan jaringan. Dengan menggunakan topologi mesh manfaat yang diperoleh dapat menambah dan mengganti klien mesh di jaringan berbasis IoT tanpa mengganggu klien mesh lainnya walau dengan konektivitas yang beragam. Sehingga dari penjelasan sebelumnya penelitian ini bertujuan melakukan perancangan "*Model Jaringan Mesh Pada Sistem Internet of Things (IoT) Untuk Pemantauan Lingkungan*" dengan kemampuan memantau kondisi lingkungan pada sistem IoT yang diharapkan mampu mendapatkan data yang variatif dari beberapa titik yang dibangun dengan menerapkan topologi mesh.

Penelitian ini menerapkan beberapa hal berkaitan tentang uji sistem dengan menerapkan 2 Skenario topologi mesh serta menggunakan parameter perngukuran data uji throughput dan latency dengan size record 8-1024 bytes dan didukung oleh sensor data untuk memantau dan menjadi parameter pengukuran pemantauan lingkungan guna menguji kinerja *broker/master/node*

Berdasarkan hasil penelitian dan pembahasan dapat disimpulkan bahwa penggunaan 2 skenario masing-masing dapat mendistribusikan data sensor sehingga berhasil melakukan broadcast data yang masuk ke server pada setiap skenario. Skenario I diperoleh hasil rata-rata throughput publish sebesar 20.75 Mbps, throughput subscriber sebesar 20.86 Mbps dan latency sebesar 6.54 ms. Serta skenario 2 dengan rata-rata throughput publish sebesar 20.75 Mbps, throughput subscriber sebesar 20.86 Mbps dan latency sebesar 8.58 ms

Kata Kunci: *Topologi Mesh, Iot, Throughput, Latency*

BAB I PENDAHULUAN

1.1 Latar Belakang

Dominasi jumlah penduduk di Indonesia meningkat setiap tahunnya. Angka tersebut meningkat pada 2025 hingga mencapai 170,4 juta jiwa atau 59,3% dari total penduduk Indonesia yang sebesar 287 juta jiwa (Jayani, 2020). Kementrian Lingkungan Hidup dan Kehutanan (KLHK) Memiliki sistem pengukuran kualitas lingkungan di beberapa kota (Kementrian Lingkungan Hidup dan Kehutanan, 2016), namun pengukuran yang dilakukan oleh KLHK hanya pada satu titik untuk satu kota, sehingga hal ini tidak bisa menjadi acuan kualitas lingkungan untuk seluruh lokasi yang ada.

Topologi *Mesh* merupakan jalur komunikasi dimana masing-masing node dapat berkomunikasi dengan yang lain. Dalam sebuah jaringan *mesh*, node mempertahankan jalur komunikasi untuk kembali ke gateway, sehingga jika salah satu node router down, secara otomatis router akan dilewatkan melalui jalur yang berbeda. (Pascale et al., 2018b) Teknologi ini memiliki kemungkinan untuk bersatu dan bagaimana mengintegrasikan jaringan *mesh* ke dalam jaringan IoT yang ada untuk berpotensi membuat perbedaan di era baru (Haseeb et al., 2020). Integrasi klien mesh dengan Internet of Things (IoT) telah menjadi sangat penting untuk menghubungkan mesin dan mencapai jangkauan cepat dengan biaya jaringan minimum (Jiang et al., 2021).

Arsitektur IoT menghubungkan semua perangkat komunikasi, baik statis maupun bergerak yang baik diaplikasikan pada pengembangan di berbagai bidang

jaringan. Selain itu, karena sifat dinamis jaringan mesh nirkabel, ini membawa dampak yang signifikan pada aplikasi berbasis IoT terutama kemampuan beradaptasi dan jangkauan jaringan dengan konektivitas yang beragam. Manfaat utama menggunakan jaringan mesh nirkabel adalah untuk menambah atau mengganti klien mesh di jaringan berbasis IoT yang ada tanpa mengganggu klien mesh lainnya sehingga sebagian besar aplikasi yang dikembangkan dengan integrasi klien mesh nirkabel dan perangkat IoT bersifat heterogen untuk algoritme, cakupan, kinerja pengiriman. Penerapan sistem IoT untuk pemantauan lingkungan menggunakan jaringan mesh ini telah dilakukan oleh peneliti sebelumnya. Sistem pemantauan lingkungan dengan open source jaringan mesh nirkabel dengan modul milik (Lin et al., 2015) sebagian lokasi yang diberikan node pada setiap lokasi data yang dihasilkan dari lokasi ini harus direlai oleh node lain sehingga menyebabkan tidak ada garis pandang antara sink antar lokassi. Sedangkan Penerapan yang sama yang digunakan peneliti (L.Karunia et al., 2019) menyatakan bahwa sistem monitoring yang menggunakan mikroserver (hardware yang berfungsi sebagai server atau sink) dapat menghindari tabrakan data dan metode DSR yang digunakan hanya terbatas pada penggunaan *cache* rule. Sehingga dari beberapa uraian dari diatas, penelitian ini bertujuan melakukan perancangan **MODEL JARINGAN MESH PADA SISTEM *INTERNET OF THINGS* (IOT) UNTUK PEMANTAUAN LINGKUNGAN** untuk merancang sebuah skenario topologi mesh dengan kemampuan memantau kondisi lingkungan pada sistem IoT untuk pemantauan lingkungan sistem IoT diusulkan mencakup desain dan pengembangan 9 node berkemampuan Wi-Fi individu papan NodeMCU ESP8266. Sistem tersebut

akan berinteraksi dengan periferal input/output dan sensor sehingga dapat berinteraksi satu sama lain dengan mencakup 2 penggunaan skenario pengujian.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah yang telah dipaparkan, maka dirumuskan masalah sebagai berikut:

1. Bagaimana membangun model jaringan *mesh* pada sistem *internet of things* untuk pemantauan lingkungan?
2. Bagaimana transmisi pengirim dan penerimaan sensor dan kinerja broker dari setiap skenario pengujian *mesh* dalam mengirimkan data pengukuran ke *server*?

1.3 Ruang Lingkup Penelitian

Agar penelitian ini tidak menyimpang dari tujuan penulisannya dan permasalahan yang dicakup tidak meluas maka pada skripsi ini akan dibatasi dengan hal-hal sebagai berikut:

1. Penelitian ini mencakup pembuatan sistem pemantauan pada skenario perkotaan dengan menggunakan topologi *mesh* dengan memanfaatkan protokol *MQTT* sebagai transmisi data
2. Penelitian ini berfokus pada wilayah perkotaan sebagai bentuk implementasi jaringan mesh pada pemantauan lingkungan.

3. Parameter pengukuran data dalam penelitian hanya meliputi sensor pengukuran suhu dan kelembapan, kadar gas *CO2* pada lingkungan dan intensitas cahaya

1.4 Tujuan Penelitian

Tujuan penelitian ini untuk:

1. Dapat mengimplementasikan topologi *mesh* dalam 2 skenario sebagai bentuk pemantauan lingkungan
2. Dapat melakukan pengukuran data dari setiap masing-masing skenario serta performa jangkauan jarak antar node *mesh* yang telah dibangun

1.5 Manfaat Penelitian

Penelitian ini mengandung beberapa manfaat yaitu:

1. Sistem yang dikembangkan dalam penelitian ini diharapkan dapat membangun sistem pemantauan pada menggunakan topologi *mesh*
2. Sistem yang dikembangkan diharapkan dapat memperoleh data yang variatif dan dapat dikustomasi dengan menentukan rentang waktu data yang diambil.

BAB II TINJAUAN PUSTAKA

2.1 Lingkungan Hidup

Indonesia menempati peringkat sepuluh besar negara Perkotaan terbesar di dunia dari tahun 1990-2014 dan memiliki populasi perkotaan terbesar kedua di Asia Timur setelah China setiap satu persen peningkatan pangsa penduduk perkotaan berkorelasi dengan peningkatan rata-rata 6-10 persen di beberapa negara Asia berpenghasilan menengah seperti China, Thailand, Vietnam, dan India. Di Indonesia, tingkat peningkatan Perkotaan yang serupa menghasilkan peningkatan PDB per kapita kurang dari 2 persen (BPIW, 2019)

a) Polusi Udara

Penyumbang pencemaran terbesar di Indonesia yaitu polusi udara kendaraan bermotor, mengingat dalam kurun waktu 10 tahun terakhir, telah terjadi lonjakan jumlah polusi kendaraan yang kian pesat, khususnya oleh pertambahan kendaraan umum terdistribusi di daerah perkotaan (Marlita et al., 2019) seperti yang ditujukan pada Tabel 2.1

Tabel 2.1 Jumlah Kendaraan 2017-2019

Jenis Kendaraan Bermotor	Perkembangan Jumlah Kendaraan Bermotor Menurut Jenis (Unit)		
	2017	2018	2019
Mobil Penumpang	13.968,202	14.830,698	15.592,419
Mobil Bis	213,359	222,872	231,569
Mobil Barang	4.540,902	4.797,254	5.021,888
Sepeda motor	100.200,245	106.657, 952	112.771,136
Jumlah	118.922,708	126.508,776	133. 617,012

Sumber: Kepolisian Republik Indonesia, 2019

b) Polusi Cahaya

Polusi cahaya adalah efek samping dari peradaban industri dan termasuk sebagai polusi udara. Sumbernya termasuk pencahayaan eksterior dan interior bangunan, iklan, properti komersial, kantor, pabrik, lampu jalan yang diterangi. studi kasus menunjukkan bahwa beberapa bentuk iluminasi berlebihan merupakan energi pemborosan. Pada skala global, sekitar 19% dari semua listrik yang digunakan menghasilkan cahaya pada malam hari (Rajkhowa, 2012) Umumnya pengaruh tersebut didominasi oleh angkutan barang sedangkan angkutan penumpang hanya memberikan pengaruh yang kecil, namun pengaruh tersebut dapat bertambah besar apabila volume lalu lintas padat di suatu jalan. Untuk besaran pengaruh kinerja lalu lintas terhadap kualitas udara ambien sebesar 28,07% dan sisanya di pengaruhi faktor lain. Hasil tersebut cukup besar untuk satu parameter pencemar udara di perkotaan (Indrayani, 2018)

c) Suhu dan Kelembapan

Berkurangnya lahan hijau daerah perkotaan terjadi karena konversi Ruang Terbuka Hijau (RTH), dan meningkatnya jumlah kendaraan yang mengakibatkan CO2 berkurang dan kuantitas serta kualitas O2 yang dihasilkan menjadi menurun (Tjaronge, 2016). Suhu dan kelembaban udara menentukan kenyamanan suatu area. Beberapa penelitian telah menyatakan bahwa pengaruh parameter iklim terhadap kenyamanan manusia. Peningkatan suhu di permukaan bumi ini terjadi karena bertambahnya zat

karbondioksida dari polusi udara yang disebabkan dari kendaraan atau hasil dari aktivitas pabrik, peningkatan gas freon, selain itu karena sinar matahari yang memancar ke bumi memantul kembali lagi ke atmosfer (Surakarta, 2010)

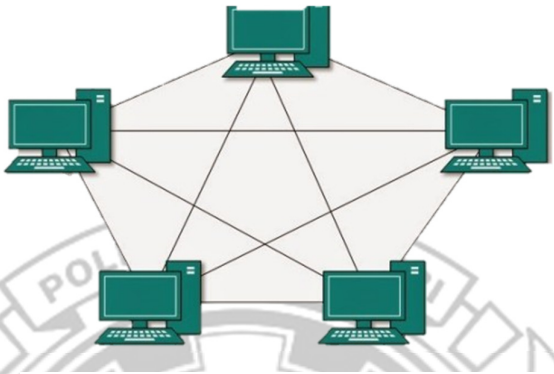
2.2 Topologi Jaringan

Topologi jaringan adalah salah satu blok bangunan utama untuk komunikasi data. Mereka menggambarkan bagaimana entitas jaringan secara langsung saling berhubungan satu sama lain dan dengan demikian menentukan bagaimana informasi dapat mengalir. Struktur hubungan simpul seperti itu dapat dibangun di atas lapisan yang berbeda yang menghasilkan topologi fisik atau logis. Yang pertama akan dibangun saat menghubungkan perangkat dengan media fisik. Di atas struktur ini, pertukaran data dapat diatur melalui jaringan dan lapisan aplikasi menciptakan topologi logis atau overlay (Wählisch, 2014)

2.2.1 Topologi *mesh*

Topologi yang menghubungkan semua komputer secara penuh. Pada hampir semua teknologi Wireless Network menggunakan Topologi Mesh. (Hestylesta, 2009) Topologi mesh merupakan bentuk hubungan yang ada antar perangkat dimana setiap perangkat yang ada akan saling terhubung langsung dengan perangkat lainnya yang ada di dalam satu jaringan tersebut. Pada topologi mesh, setiap perangkat yang ada dapat berkomunikasi langsung dengan perangkat lainnya dikarenakan perangkat akan saling terhubung yang dikenal dengan *dedicated links*.

Komunikasi yang terjalin pada topologi mesh biasanya berjalan cepat dan dapat digunakan untuk membangun jaringan yang skalanya tidak terlalu besar.



Gambar 2.1 Topologi Mesh

Sumber : https://www.nesabamedia.com/topologi-mesh/

Topologi ini biasanya timbul akibat tidak adanya perencanaan awal ketika membangun suatu jaringan. Karena tidak teratur maka kegagalan komunikasi menjadi sulit dideteksi, dan ada kemungkinan boros dalam pemakaian media transmisi. Topologi ini menerapkan hubungan antar sentral secara penuh. Jumlah saluran yang harus disediakan untuk membentuk jaringan Mesh adalah jumlah sentral dikurangi 1. Tingkat kerumitan jaringan sebanding dengan meningkatnya jumlah sentral yang terpasang. Disamping kurang ekonomis juga relatif mahal dalam pengoperasiannya. Topologi ini merupakan teknologi khusus yang tidak dapat dibuat dengan pengkabelan, karena sistem yang rumit. Namun dengan teknologi wireless, topologi ini sangat memungkinkan untuk diwujudkan (Green, 2018)

Topologi mesh suatu bentuk hubungan antra perangkat/node dimana setiap perangkat terhubung secara langsung ke perangkat lainnya. Tiap-tiap node dalam

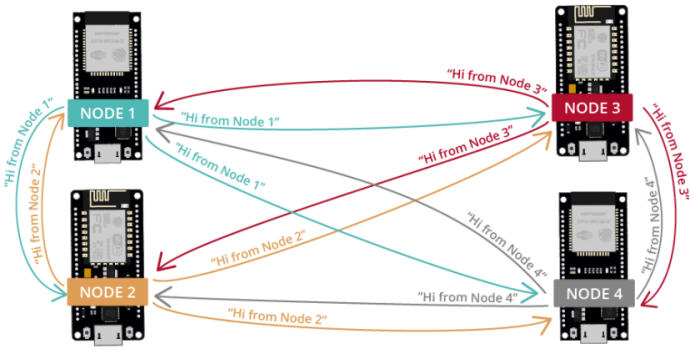
topologi mesh tidak hanya berfungsi sebagai penerima data untuk dirinya sendiri namun juga sebagai penyedia data untuk perangkat/node yang lainnya. Memiliki sifat Robust, yaitu apabila terjadi gangguan pada koneksi komputer A ddengan komputer B karena rusaknya kabel koneksi (links) antara A dan B, maka gangguan tersebut tidak akan memengaruhi koneksi komputer A, dengan komputer lainnya. Topologi *mesh* merupakan bentuk topologi yang sangat cocok dalam hal pemilihan rute yang banyak. Hal tersebut berfungsi sebagai jalur *backup* pada saat jalur lain mengalami masalah.

Node jaringan adalah bagian dari infrastruktur mereka buat, dan tujuan utamanya adalah untuk melakukan perutean tugas. Mobilitas node mesh terbatas atau nol, dan kemampuan pemrosesan, memori, dan bandwidth biasanya lebih tinggi daripada jaringan khusus tradisional node. Selain itu, kebutuhan daya mesh jaringan umumnya kurang ketat daripada nirkabel ad-hoc jaringan. Saat menghubungkan node masing-masing ke masing-masing,

2.3 Protokol

2.3.1 ESP-MESH

ESP-MESH adalah jaringan komunikasi nirkabel dengan node yang diatur dalam topologi mesh menggunakan fitur AP-STA simultan pada SoC Espressif. Ini menyediakan jaringan self-forming dan self-healing, dengan kemudahan penyebarannya. Topologi jaringan ESP-WIFI-MESH dapat menskalakan hingga 1000 node di area yang luas, tanpa memerlukan dukungan infrastruktur Wi-Fi khusus.



Gambar 2.2 ESP-MESH

Sumber: https://randomnerdtutorials.com/esp-mesh-esp32-esp8266-painlessmesh/#1
Diakses 08 Agustus 2021

2.3.2 ESP-NOW

ESP-NOW adalah protokol sebuah protokol peer-to-peer Protokol ini tidak memerlukan router untuk jaringan hirarki. ESP-NOW memberikan perangkat kemampuan berkomunikasi dengan metode broadcast, unicast, dan multicast. ESP-NOW memiliki fitur seperti komunikasi unicast yang bisa dienkripsi maupun tidak dienkripsi, bisa dicampur antara perangkat yang dienkripsi dan tidak dienkripsi, pengiriman data hingga 250 byte, dan fungsi callback pengiriman bisa diatur untuk mengetahui apakah pengiriman data berhasil atau gagal. ESP-NOW juga memiliki beberapa limitasi seperti node yang terenkripsi terbatas, yaitu 10 untuk mode station, 6 untuk mode softAP atau softAP+station (Espressif, 2019)

2.3.3 *Painless Mesh*

PainlessMesh merupakan sebuah *library* yang berfungsi untuk membuat jaringan *Mesh* (jala) sederhana padaESP8266 secara otomatis tanpa perlu membuat

struktur dari topologi jaringan tersebut. *PainlessMesh* menggunakan JSON sebagai basis dari pertukaran data yang dilakukan antar*node*. *PainlessMesh* menggunakan JSON karena mudah dibaca dan dipahami, selain itu juga mudah diintegrasikan dengan javascript *front-end* maupun web aplikasi lainnya (Martin, G. 2019)

2.3.4 Message Queue Telementry Transport (MQTT)

MQTT merupakan protokol berbasis TCP yang dikembangkan oleh International Business Machines (IBM) dengan pola interaksi *publish/subscibe*, yang terdiri dari satu broker server dan dua jenis klien yang disebut *Publisher* (*Publish Client*) dan *Subscriber* (*Subcribe Client*). Broker bertindak sebagai perantara pesan yang dikirim antara *Publish Client* dan *Subscriber Client* untuk topik tertentu (Rizzardi et al., 2016). Prinsip inilah yang membuat protokol ini ideal untuk diaplikasikan pada komunikasi machine-to-machine (M2M) atau Internet of Things dan untuk aplikasi mobile dimana bandwidth dan kapasitas baterai terbatas

2.3.4.1 *Mosquitto*

Untuk penelitian ini, broker yang diterapkan ialah *Mosquitto*. *Mosquitto* sendiri tersedia implementasi antara klien dan server yang sesuai dengan standar dari protokol pesan MQTT. Ada tiga bagian *Mosquitto* yaitu server utama *Mosquitto*, utilitas klien *mosquitto_pub* dan *mosquitto_sub* yang merupakan salah satu metode untuk berkomunikasi dengan sebuah server MQTT (Light, 2017)

2.4 Mikrokontroller

Mikrokontroller merupakan sebuah sistem komputer fungsional yang di letakkan dalam papan elektronik yang berukuran mikro atau kecil. Di dalam

mikrokontroller terdapat sebuah prosesor, memori, serta komponen *input* dan *output*. Dengan begitu, mikrokontroller adalah suatu alat elektronika digital yang memiliki *input* serta *output* yang kendalinya dapat diprogram ulang dengan suatu cara khusus. Ada banyak pilihan mikrokontroller yang telah terdapat modul jaringan agar dapat langsung terhubung ke jaringan internet sehingga dapat di implimentasikan untuk membuat peralatan berbasis IoT/WSN, contohnya NodeMCU ESP8266.

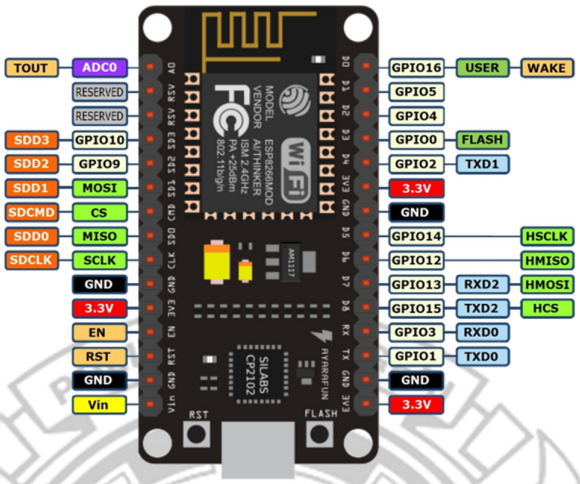
2.4.1 NodeMCU ESP-8266

NodeMCU ESP-8266 Merupakan *single board microcontroller* yang ditenagai memori 128kBytes, tempat penyimpanan 4Mbytes, dan bersumber daya dari USB. Mikrokontroller ini memiliki modul WiFi dan *firmware* menggunakan bahasa pemrograman LUA atau Arduino IDE (Fajrika hadnis sPutra et al., 2018)

Fitur-fitur:

a) Versi: DevKit v1.0
b) *Breadboard Friendly*
c) Ringan dan ukuran kecil.
d) 3.3V dioperasikan, dapat didukung USB.
e) Menggunakan protokol nirkabel 802.11b / g / n.
f) Kemampuan konektivitas nirkabel built-in.
g) Built-in PCB antenna pada chip ESP-12E.
h) Mampu PWM, I2C, SPI, UART, 1-kawat, 1 pin analog.
i) Menggunakan modul antarmuka Komunikasi Serial CP2102 USB.
j) Arduino IDE kompatibel (diperlukan manajer papan ekstensi).
k) Mendukung Lua (node.js) dan bahasa pemrograman Arduino C.

1. PinOut *Diagram*



Gambar 2 3 PinOut Diagram
Sumber: (EINSTRONIC, 2017)

2.5 Sensor Data

Sensor adalah alat untuk mendeteksi atau mengukur sesuatu, yang digunakan untuk mengubah variasi mekanis, magnetis, panas, sinar dan kimia menjadi tegangan dan arus listrik. Dalam lingkungan sistem pengendalian dan robotika, sensor memberikan kesamaan yang menyerupai mata, pendengaran, hidung, lidah yang kemudian akan diolah oleh *controller* sebagai otaknya (v. M. buyanov, 2017)

2.5.1 Sensor DHT 22

Sensor DHT22 adalah Sensor Suhu & Kelembaban DHT22 dilengkapi dengan sensor suhu & kelembaban kompleks dengan output sinyal digital yang dikalibrasi. Sensor ini memiliki kalibrasi akurat dengan kompensasi suhu ruang penyesuaian dengan nilai koefisien tersimpan dalam memori OTP terpadu. Sensor DHT22 memiliki rentang pengukuran suhu dan kelembaban yang luas, DHT22

mampu mentransmisikan sinyal keluaran melewati kabel hingga 20 meter sehingga sesuai untuk ditempatkan di mana saja. Sensor ini mencakup pengukuran kelembaban tipe resistif komponen dan komponen pengukuran suhu NTC, dan terhubung ke kinerja tinggi Mikrokontroler 8-bit, respons cepat, anti-interferensi kemampuan dan efektivitas biaya (Sensor, n.d.)

Tabel 2.2 Spesifikasi Sensor DHT 22

Detail	Spesifikasi
Rentang deteksi suhu	-40 - + 80° Celsius (akurasi +/- 0,5°C)
Resolusi Sensivitas	0,1%RH; 0,1°C
Histeresis kelembaban	+- 0,3% RH
Temperature Range	0°C to 50°C
Catu Daya	3,3 – 6 Volt DC

Sumber : https://components101.com/sensors/dht11-temperature-sensor *diakses 08 Agustus 2021*

2.5.2 Sensor MQT135

Sensor gas MQ135 adalah SnO2, yang memiliki konduktivitas lebih rendah di udara bersih. Ketika target gas yang mudah terbakar ada, konduktivitas sensor lebih tinggi seiring dengan kenaikan konsentrasi gas. Sensor gas MQ135 juga memiliki sensitivitas tinggi terhadap uap Amoniak, Sulfida dan Benz, juga sensitif terhadap asap dan gas berbahaya lainnya. (Winseng, 2015)

Tabel 2.3 Spesifikasi Sensor MQT135

Spesifikasi
Fast response and High sensitivity Stable and long life Operating Voltage is +5V Detect/Measure NH3, NOx, alcohol, Benzene, smoke, CO2, etc. Analog output voltage: 0V to 5V Digital output voltage: 0V or 5V (TTL Logic) Preheat duration 20 seconds Can be used as a Digital or analog sensor The Sensitivity of Digital pin can be varied using the potentiometer

Sumber : https://components101.com/sensors/mq135-gas-sensor-for-air-quality diakses 08 Agustus 2021

2.5.3 Sensor TSL2561

Sensor luminositas TSL2561 adalah sensor cahaya digital canggih, ideal untuk digunakan dalam berbagai situasi cahaya. Dibandingkan dengan sel CdS berbiaya rendah, sensor ini lebih presisi, memungkinkan perhitungan Lux yang tepat dan dapat dikonfigurasi untuk rentang gain/waktu yang berbeda untuk mendeteksi rentang cahaya dengan cepat. bagian terbaik dari sensor ini adalah mengandung dioda spektrum inframerah dan full spectrum (Adafruit et al., 2018)

Tabel 2.4 Spesifikasi Sensor TSL2561

Detail	Spesifikasi
Interface	12C -VDD
Programmable	Gaain, integration time, interrupt
Temperature Range ^{0}C	-30 to 70
Package	(6-pin) CL, FN, CS, T

Sumber : https://ams.com/tsl2561 diakses 08 Agustus 2021

2.6 Paramater Pengukuran Data

2.6.1 Throughput

Throughput jaringan adalah jumlah data yang berhasil dipindahkan dari satu tempat ke tempat lain dalam periode waktu tertentu, dan biasanya diukur dalam bit per detik (bps). Detail kategori *throughput* dijabarkan dalam Tabel 2.5

Tabel 2.5 Kategori *Throughput*

Kategori *Throughput*	***Throughput***	**Indeks**
Sangat Baik	>2.1 mbps	4
Baik	1,200 kbps – 2.1 mbps	3
Cukup	700 – 1,200 kbps	2
Kurang Baik	338 – 700 kbps	1
Buruk	0 – 338 kbps	0

Sumber: (TIPHON, 2002)

Untuk mengukur nilai *throughput* digunakan:

$$throughput = \frac{Ukuran\ data\ yang\ diterima}{\text{waktu pengiriman data}} \quad \cdots\cdots\cdots\cdots\cdots\cdots \quad (1)$$

2.6.2 Latency

Latency merupakan waktu yang dibutuhkan data untuk menempuh jarak dari asal ke tujuan dan salah satu tolak ukur untuk mengukur performa jaringan. *Latency* digunakan untuk menghitung waktu yang dibutuhkan data untuk menempu jarak dari asal ke tujuan. Detail *Latency* dijabarkan dalam Tabel 2.6

Tabel 2.6 Kategori Latency

Kategori *Latency*	Besar *Latency*	Indeks
Sangat Baik	< 150 ms	4
Baik	150 ms s/d 300	3
Sedang	300 ms s/d 450	2
Buruk	> 450 ms	1

Sumber:(TIPHON, 2002)

Untuk mengukur nilai Latency digunakan:

$$Rata - rata\ latency = \frac{Total\ Latency}{\text{Total Paket yang diterima}} \quad \cdots\cdots\cdots\cdots\cdots\cdots \quad (2)$$

2.7 Server

2.7.1 Raspberry Pi

Raspberry Pi adalah jenis komputer *Single Board Circuit /SBC* yang berukuran sebesar kartu kredit. *Rasberry Pi* dapat digunakan untuk berbagai keperluan seperti komputasi, *spreadsheet*, *game* bahkan untuk keperluan memutar video). Pada penelitian ini *server* yang digunakan adalah *Rasberry Pi* 4 model b yang berfungsi sebagai sumber perangkat untuk melakukan proses manajemen data

dan penyedia layanan *web service* bagi *user* yang ingin melakukan monitoring. Berikut spesifikasi *Raspberry Pi* 4 model b yang digunakan:

Tabel 2.7 Spesifikasi Raspberry Pi 4

Spesifikasi	Raspberry Pi 4
Processor	Quad core Cortex-A72 (ARM v8) 64-bit SoC
RAM Size	4 GB RAM
Wireless	2.4 GHz and 5.0 GHz IEEE 802.11ac
Bluetooth	Bluetooth 5.0, BLE
Ethernet	Gigabit Ethernet
USB Ports	- USB Ports 2.0 - USB Ports 3.0
HDMI Ports	Micro-HDMI
Display Port	MIPI DSI Display Port
Camera Port	MIPI CSI Display Port
Power	- 5V DC via USB-C connector (minimum 3A*) - 5V DC via GPIO header (minimum 3A*)

Sumber : https://www.raspberrypi.org diakses 25 Juni 2022

BAB III METODOLOGI PENELITIAN

3.1 Tempat dan Waktu Penelitian

Penelitian ini dilaksanakan di LAB CNAP Kampus I Politeknik Negeri Ujung Pandang, pada bulan April 2022 hingga Oktober 2022.

3.2 Alat dan Bahan

Perangkat penelitian yang digunakan pada penelitian meliputi perangkat keras dan perangkat lunak, sistem operasi dan aplikasi pendukung yang digunakan pada penelitian ini.

3.2.1 Perangkat Keras

Tabel 3.1 Kebutuhan Perangkat Keras

No	Perangkat Keras	Deskripsi
1	Laptop a. Processor Intel(R) Core(TM) i3-8145U CPU @ 2.10GHz 2.30 GHz b. Memory 8Gb c. Jumlah 1 Buah	Digunakan untuk merancang sistem dan mengedit *script*.
2	ESP 8266 (12 Buah)	Mikrokontroler untuk mengontrol dan mengendalikan serta memantau perangkat dari jarak jauh melalui jaringan *Wireless Fidelity* (Wi-Fi). Serta digunakan untuk mengirim serta menerima data sensor maupun data konfigurasi

3	Sensor a. 3 Sensor DHT 22 (Suhu dan Kelembaban) b. 3 Sensor MQT135 (Mengukur Kadar Gas CO2 Pada Lingkungan) c. 3 Sensor TSL2561 (Mengukur Intesitas Cahaya)	Alat yang digunakan sebagai media pengambilan data yang dikirimkan kedatabase melalui *ESP-8266*
4	Raspberry Pi	Alat yang digunakan sebagai sebuah perangkat komputer yang memungkinkan untuk memonitor dan menprogram seperti bahasa *python*

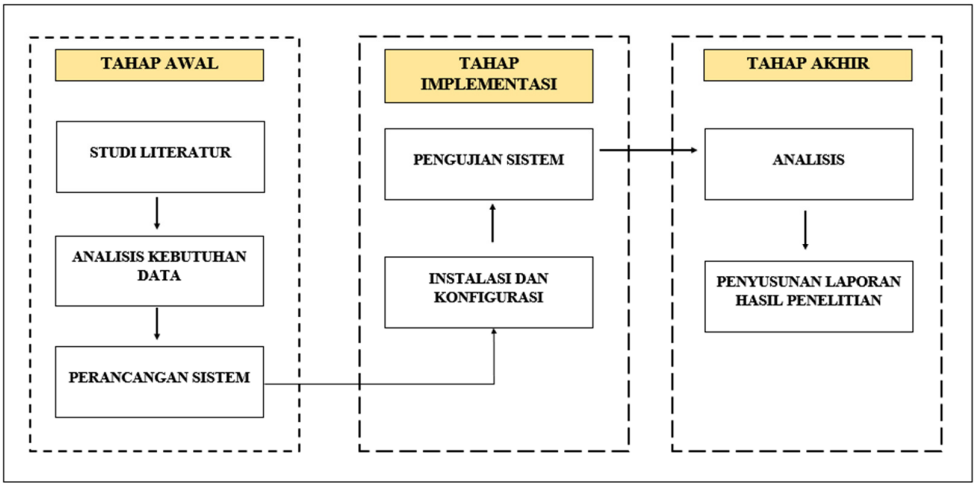
3.2.2 Perangkat Lunak

Tabel 3.2 Kebutuhan Perangkat Lunak

No	Perangkat Lunak	Deskripsi
1	*Arduino Integrated Developtment Environment* (IDE)	Aplikasi editor untuk membuat, membuka, mengedit, dan juga memvalidasi kode serta untuk di *upload* ke papan Arduino

3.3 Prosedur Penelitian

Prosedur penelitian digunakan agar penelitian yang dilakukan lebih terarah dan terstruktur sehingga hasil yang diperoleh sesuai dengan tujuan penelitian. Adapun prosedur penelitian dijelaskan pada Gambar 3.1



Gambar 3.1 Prosedur Penelitian

3.3.1 Studi Literatur

Tahapan awal yakni studi literatur dan menentukan cakupan sistem yang akan dibuat. Pada tahap ini dilakukan studi literatur dengan mempelajari, membaca, dan mencatat literatur dari beberapa buku maupun jurnal, serta mencari kebutuhan sistem yang akan dibuat maupun artikel dengan narasumber yang jelas dan terpercaya serta mencari kebutuhan yang diperlukan sistem.

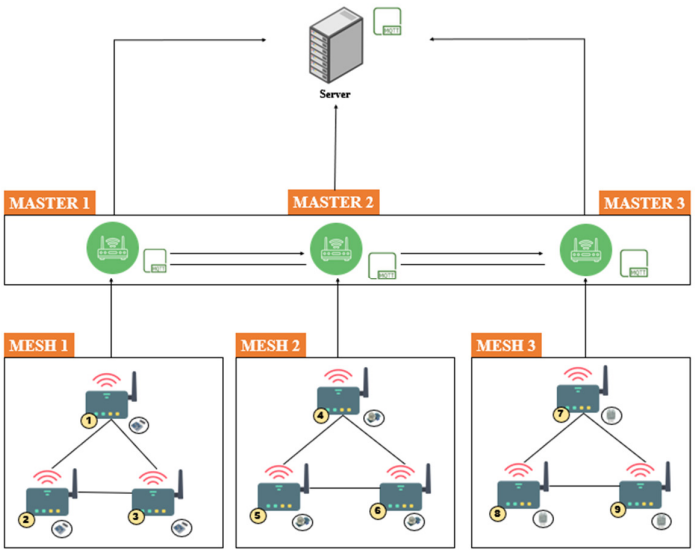
3.3.2 Analisis Kebutuhan Data

Pada penelitian ini data yang diperlukan untuk melakukan pengimplentasian mesh untuk pemantauan lingkungan dalam skenario perkotaan ialah data pada

perangkat mikrokontroller yang terdiri dari data suhu, kelembaban, gas *Co2*, intensitas cahaya matahari. Adapun rancangan sistem secara keseluruhan dari penelitian ini dapat dilihat pada Gambar 3.2

3.3.3 Perancangan Sistem

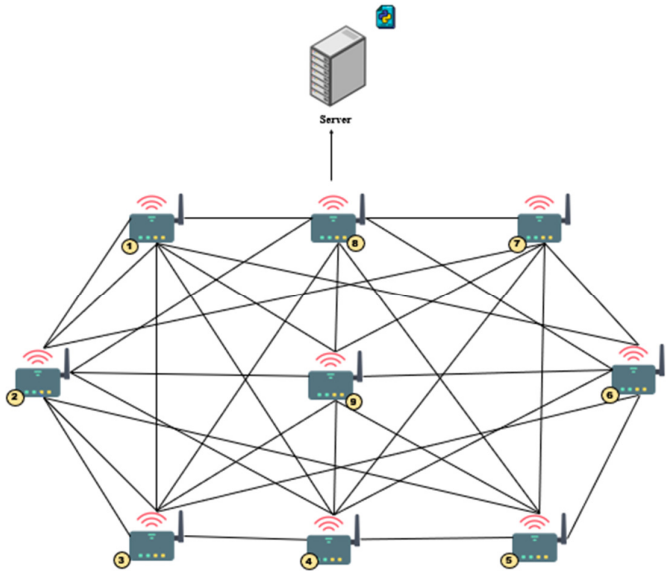
Penyusunan skenario pada penelitian ini didasarkan untuk memberikan gambaran tentang tahapan penelitian yang akan dilaksanakan. Penelitian ini menggunakan dua pengimplementasian skenario. untuk perancangan dua skenario pengujian pada pemantauan lingkungan ini disajikan secara skematis pada Gambar 3.1 dan Gambar 3.2



Gambar 3.2 Perancangan Skenario Pengujian 1

Perancangan skenario mesh untuk skenario satu pada Gambar 3.2 diterapkan pengukuran dengan titik masing-masing mesh memiliki masing-masing tiga node. Masing-masing mesh diatas hanya berkomunikasi sesama node yang dimiliki

sehingga antar mesh lainnya tidak melakukan komunikasi ke mesh lainnya sehingga digunakan pengaplikasian tiga perangkat master yang akan menjadi tempat akhir pengiriman data dari perangkat mesh dan sebagai wadah penyimpanan data untuk melakukan *broadcast* data sensor yang tadinya diterima oleh perangkat mesh ke pengiriman master lain. Master yang telah menerima data sensor dari perangkat mesh lalu dilakukan pengiriman ke server

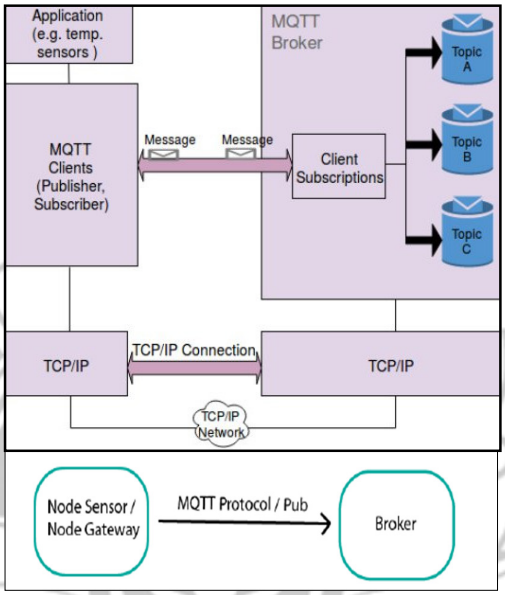


Gambar 3.3 Perancangan Skenario Pengujian 2

Perancangan skenario mesh untuk skenario dua pada Gambar 3.3 diterapkan pengukuran dengan titik jangkauan node yang mampu terhubung satu sama lain tanpa mengaplikasikan perangkat master seperti yang ada pada skenario pertama. Skenario dua memanfaatkan sembilan node dan sembilan sensor data sebagai pengaplikasian pemantauan lingkungan yang variatif dengan penggunaan satu server sebagai tempat akhir untuk penerimaan data ke server.

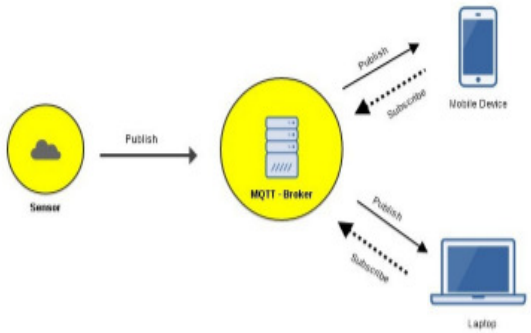
1) Proses Komunikasi

Proses komunikasi ini menggunakan protokol komunikasi yaitu Protokol MQTT.



Gambar 3.4 Gambaran Umum TCP/IP

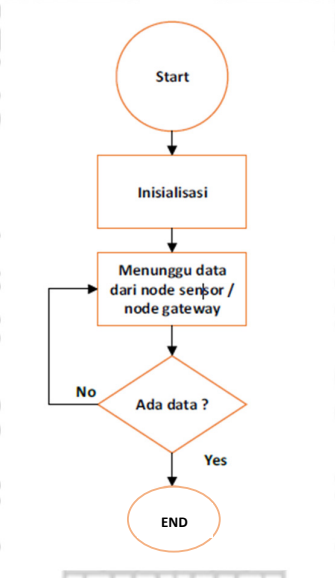
Proses ini diharapkan sebagai penghubung untuk mengirimkan dan membaca data sensor dari master untuk skenario 1 dan masing-masing node untuk skenario 2 ke server sebagai penyimpanan data pengukuran. Adapun gambaran umum protocol MQTT untuk melakukan *publish/subscribe* mengirim dan membaca data sensor dapat dilihat pada Gambar 3.4 dan 3.5



Gambar 3.5 Gambaran Umum MQTT

2) Proses data di *Server*

Skenario pengujian I *server* akan menunggu data dari *pendistibusian* oleh masing-masing jenis perangkat dimana master untuk skenario 1 dan node-node untuk skenario 2, jika terjadinya kegagalan pengeriman data, maka *server* akan menunggu kembali pengiriman data dari node & *master*. Jika *server* berhasil menerima data dari node & *master* (Temperature, Humidity, Gas *CO2*, Infrared, Full Spectrum, date and time) Proses data di *server* dapat dilihat pada Gambar 3.6



Gambar 3.6 Proses data *publish*

3.3.4 Pengujian Sistem

Pada bagian pengujian akan menerangkan beberapa hal berkaitan tentang uji sistem berdasarkan implementasi dari dua skenario. Jenis pengujian yang dilakukan dengan menggunakan beberapa pengukuran data yang dijabarkan pada bab 4 pengujian sistem serta uji kinerja *broker* dengan menggunakan parameter

pengukuran data dengan menerapkan uji *throughput* dan *latency*. Dengan spesifikasi beberapa tahap pengujian yaitu:

a) Spesifikasi pengujian *throughput* terhadap jumlah *size* per *record*

Jumlah masing-masing *Node* tiap *Mesh*	Jumlah message	Size / record (*byte*)
3 Node (Skenario Pengujian 1)	10000	8, 16, 32, 64, 128, 256, 512 dan 1024 *bytes.*
9 Node (Skenario Pengujian 2)	10000	8, 16, 32, 64, 128, 256, 512 dan 1024 *bytes.*

b) Spesifikasi pengujian *latency* terhadap jumlah *size* per *record*

Jumlah masing-masing *Node* tiap *Mesh*	Jumlah message	Size / record (*byte*)
3 Node (Skenario Pengujian 1)	10000	8, 16, 32, 64, 128, 256, 512 dan 1024 *bytes.*
9 Node (Skenario Pengujian 2)	10000	8, 16, 32, 64, 128, 256, 512 dan 1024 *bytes.*

3.3.6 Analisis Sistem

Pada tahap ini dilakukan analisis terhadap kinerja hasil pengujian, selanjutnya dijadikan bahan dokumen produk akhir dalam penelitian ini. Masalah yang ditemukan dalam proses penelitian turut dijadikan sebagai data

hasil penelitian untuk memudahkan pembaca dalam mengembangkan penelitian lebih lanjut :

Analisis kinerja *throughput* dan *latency*

Throughput diukur dengan mengamati jumlah total kedatangan paket yang berhasil, yang diamati pada tujuan selama interval waktu tertentu dibagi oleh durasi interval waktu tersebut. *Latency* diukur dengan mengamati waktu yang dibutuhkan data untuk menempuh jarak dari asal ke tujuan. Tabel 3.4. menunjukkan langkah-langkah identifikasi *throughput* dan *latency*

Tabel 3.4 Identifikasi *throughput* dan *latency* menggunakan MQTTLoader

Langkah	Proses
Langkah 1	Menjalankan MQTT
Langkah 2	Mengakses topik *subscribe master* tiap *mesh*
Langkah 3	Menentukan jumlah *record* dan *size* per record
Langkah 4	Setelah menentukan jumlah *record* dan *size* dapat menjalankan **MQTTLoader** dan mengatur default configuration parameter didalam ***mqttloader.conf*** untuk *throughput* dan *latency*
Langkah 5	Amati dan catat hasil pada tabel *throughput*

BAB IV HASIL DAN PEMBAHASAN

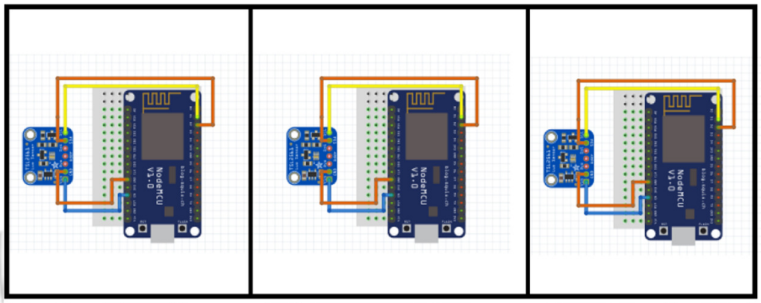
Hasil yang didapatkan dari penelitian ini berdasarkan rancangan, tahapan – tahapan pelaksanaan, dan pengujian yang sesuai dengan pokok permasalahan dan ruang lingkup penelitian. Penelitian ini berfokus pada perancangan implementasi jaringan mesh untuk pemantauan lingkungan serta penerapannya diperlukan pembuatan alat pada mikrokontroller dengan menggunakan topologi *mesh*. Parameter Pengukuran data pada penelitian ini diterapkan uji *throughput* dan *latency* untuk mengukur laju data persatuan waktu dan menghitung waktu yang ditempuh untuk kebutuhan data jarak dari asal ke tujuan. Pengujian ini dilakukan untuk mengetahui bahwa konfigurasi yang dilakukan dapat berjalan dengan semestinya dengan mengikuti tahapan pengujian sistem yang ada pada Bab III

4.1 Implementasi

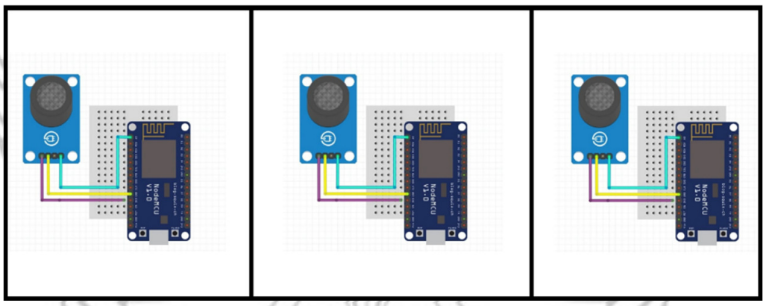
4.1.1 Rancang Bangun Perangkat ESP-*Mesh*

Rancang Bangun Perangkat *Mesh* meliputi pengujian perangkat yang telah dibuat, Pengimplementasian menggunakan 3 ESP-MESH yang setiap ESP-MESH terdiri dari *3 node* dan masing *node* nya terdiri dari 3 sensor data yang mewakili pengukuran kondisi pada lingkungan yang sejenis. Pada sensor *node 1, node 2, dan node 3* perangkat yang di gunakan adalah NodeMCU ESP8266 tipe 12-E yang dipasangkan dengan sensor *TSL2561* yang berfungsi sebagai pembaca Kualitas Cahaya. Sensor *node 4, node 5, node 6* perangkat yang digunakan adalah NodeMCU ESP8266 tipe 12-E dipasangkan dengan sensor MQ135 yang berfungsi

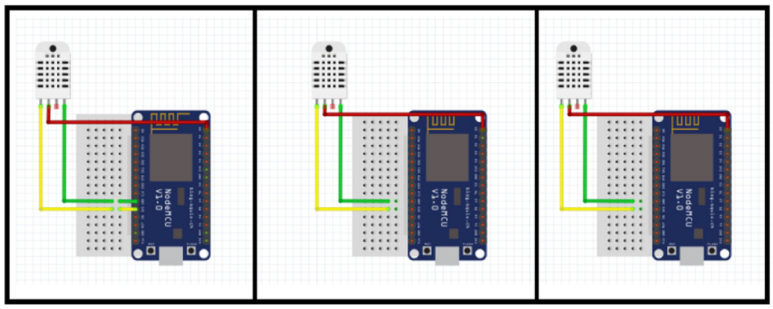
sebagai pembaca kualitas udara CO2. Dan sensor *node 7, node 8, node9* perangkat yang digunakan adalah NodeMCU ESP8266 tipe 12-E dipasangkan dengan sensor DHT22 yang berfungsi sebagai pembaca Suhu dan Kelembapan. Perkotaan. Adapun perangkat keras yang telah dibuat pada penelitian ini dapat dilihat pada Gambar 4.1 s/d Gambar 4.3



Gambar 4.1 Board NodeMCU & TSL2561



Gambar 4.2 Board NodeMCU & MQ135



Gambar 4.3 Board NodeMCU & DHT22

4.1.2 Implementasi *PainlessMesh*

Implementasi *PainlessMesh* meliputi penerimaan data dan informasi dari masing-masing *Mesh* dari seriap *node* yang masuk, *node* tersebut mengirimkan data berdasarkan pembacaan dari sensor yang ada dengan mengaplikasikan topologi *mesh.* Untuk itu diperlukan *librari PainlessMesh* untuk mendukung kinerja pengiriman dan penerimaan data pada masing-masing sensor ke board NodeMCU ESP8266 di masing-masing *Node Mesh* baik dalam Implementasi Pengujian Skenario 1 dan Skenario 2

4.1.3 Implementasi ESP-NOW

Implementasi ESP-NOW digunakan untuk pendistribusian pada skenario pengujian I yang meliputi penerimaan data dan informasi dari masing-masing Node *Mesh* dari seriap *perangkat* yang masuk, *node* tersebut mengirimkan data berdasarkan pembacaan dari sensor yang ada dengan mengaplikasikan pemanggilan port yang dibuat berdasarkan masing-masing titik node mesh agar data yang dikirimkan ke master masih berupan mesh namun saat melakukan distribusi ke sesama master sudah menggunakan esp-now sebagai transmisi datanya. Untuk itu diperlukan *librari* ESP-NOW untuk mendukung kinerja pengiriman data pada skenario pengujian 1

4.2 Pengujian

Pengujian yang dilakukan untuk masing-masing skenario satu dan dua digunakan penerapan implementasi sesuai dengan yang dijabarkan sebelumnya

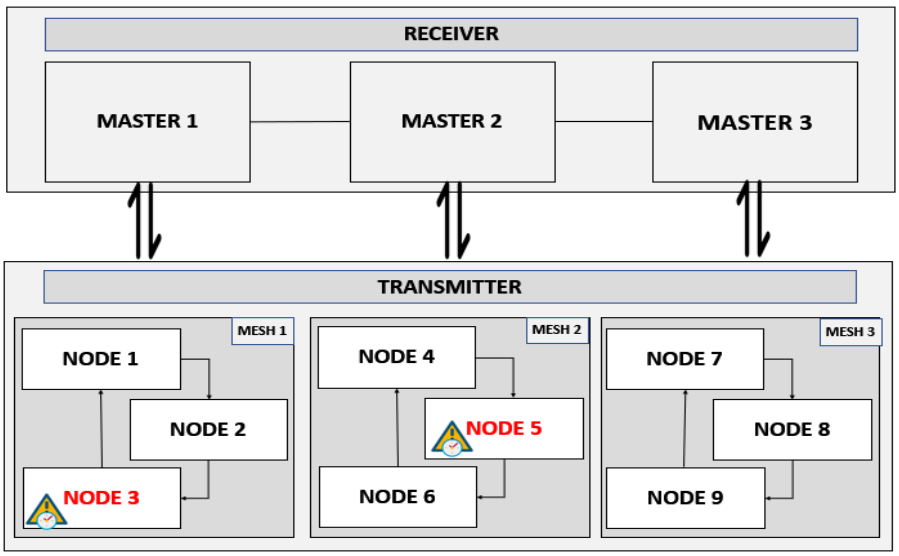
dalam Gambar 3.2 dan Gambar 3.3. Pada bagian ini dilakukan dua jenis pengujian pada masing-masing jenis skenario yaitu:

a) **Pengujian keseluruhan node transmitter**

Pengujian keseluruhan node menerapkan uji masing-masing skenario guna melihat penerapan yang dilakukan apakah mampu dijalankan pada keseluruhan node di masing-masing skenario dalam melakukan pengiriman data.

b) **Pengujian dengan dua node transmitter dalam keadaan off**

Pengujian ini untuk menguji node receiver dalam menerima data yang dikirim oleh dua node transmitter yang dalam keadaan off, serta menguji node receiver menerima hasil data dari transmitter dan mendistribusikan data ke perangkat master. Adapun node yang diujikan tertera pada Gambar 4.3

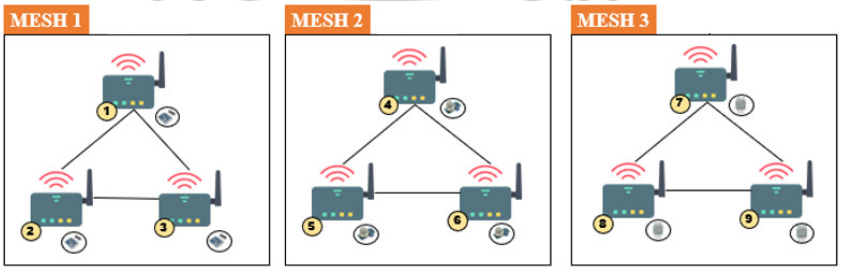


Gambar 4.3 Pengujian dua node transmitter dalam keadaan off

4.2.1 Pengujian Penerimaan Data Skenario I

a) Pengujian keseluruhan node transmitter

Pengujian penerimaan data sensor yang dikirim berupa node antar masing-masing mikrokontroller ke masing-masing pasangan *mesh*. Data sensor ini dikirim ke board masing-masing secara *continue* terus menerus menggunakan protocol *painlessmesh*.



Gambar 4.4 Pengujian keseluruhan node transmitter

Dalam pengujian ini dibutuhkan masing-masing 3 node dari masing-masing pengukuran pemantauan lingkungan. Apabila masing-masing node yang berisikan 3 node / masing-masing ESP hanya beberapa node mengirim data ke master atau salah satu node router down, secara otomatis node akan dilewatkan melalui jalur yang berbeda sehingga dalam satu mesh tersebut akan tetap melakukan pengiriman data pada sensor node lain untuk dapat terhubung ke satu sama lain dan dapat saling berkomunikasi diantara node-node dalam tiap-tiap mesh tersebut dan mendistribusikan data dari *mesh* agar dapat diteruskan ke *server*. Adapun proses penerimaan data sensor pada *mesh* dapat dilihat pada Gambar 4.5, Gambar 4.6 dan Gambar 4.7

ESP-TSL2561

```
bridge: Received from 539538257 msg={"id":2,"lux":3}
bridge: Received from 539550132 msg={"id":1,"lux":6}
bridge: Received from 539157004 msg={"id":3,"lux":2}
bridge: Received from 539538257 msg={"id":2,"lux":3}
bridge: Received from 539550132 msg={"id":1,"lux":6}
bridge: Received from 539157004 msg={"id":3,"lux":3}
bridge: Received from 539538257 msg={"id":2,"lux":3}
bridge: Received from 539550132 msg={"id":1,"lux":6}
bridge: Received from 539157004 msg={"id":3,"lux":4}
bridge: Received from 539550132 msg={"id":1,"lux":6}
bridge: Received from 539538257 msg={"id":2,"lux":3}
bridge: Received from 539157004 msg={"id":3,"lux":4}
```

Gambar 4.5 Proses Penerimaan data 3 Node ESP-TSL2561

Pada Gambar 4.5 menunjukkan hasil penerimaan data sensor antara masing-masing *node* ke dalam ESP-TSL2561 yang dimana ketiga *node* yang ada saling terkoneksi satu sama lain yang ditandai dengan hasil *received* antara node *id* dari masing-masing node yang dimiliki masing-masing *node.* Penerapan hasil jangkauan node yang telah dijabarkan pada Bab 3 Gambar 3.1 Skenario Pengujian 1 berhasil dilakukan dalam antar masing-masing *node* dan mampu melakukan pembacaan masing-masing node dalam 1 mesh tersebut. Seperti pada MESH-1 ESP TSL2561 menerima hasil dari ketiga node berdasarkan pada *id* yang merupakan nodenumber dan isi dari penginisialisasian *lux* ialah satuan pengukuran dari penerangan yang sama dengan penerangan langsung pada permukaan

ESP-MQ135

```
bridge: Received from 2829316923 msg={"id":4,"ppm":5832.201171875}
bridge: Received from 2829328707 msg={"id":6,"ppm":12866.173828125}
bridge: Received from 2829369860 msg={"id":5,"ppm":25932.822265625}
bridge: Received from 2829328707 msg={"id":6,"ppm":12932.89453125}
bridge: Received from 2829369860 msg={"id":5,"ppm":26044.4609375}
bridge: Received from 2829316923 msg={"id":4,"ppm":5832.201171875}
bridge: Received from 2829328707 msg={"id":6,"ppm":12932.89453125}
bridge: Received from 2829369860 msg={"id":5,"ppm":26044.4609375}
bridge: Received from 2829316923 msg={"id":4,"ppm":5832.201171875}
bridge: Received from 2829328707 msg={"id":6,"ppm":12932.89453125}
bridge: Received from 2829369860 msg={"id":5,"ppm":25932.822265625}
bridge: Received from 2829316923 msg={"id":4,"ppm":5832.201171875}
```

Gambar 4.6 Proses Penerimaan data 3 Node ESP-MQ135

Pada Gambar 4.6 sama halnya dengan Gambar 4.4 untuk Gambar 4.5 menunjukkan hasil penerimaan data sensor antara masing-masing *mesh* yang dimana ketiga *node* yang ada saling terkoneksi satu sama lain yang ditandai dengan hasil *received* antara node *id* dari masing-masing node yang dimiliki masing-masing *node.* Penerapan hasil jangkauan node mesh untuk ESP-MQ135 menerima hasil dari ketiga node berdasarkan pada perhitungan penerimaan pemantauan kualitas udara yaitu *id* dan isi dari penginisialisasian satuan PPM sebagai satuan pengukuran yang digunakan untuk menyatakan tingkat konsentrasi polutan di udara

ESP-DHT22

```
bridge: Received from 2829692915 msg={"id":7,"hum":76.699996948242188,"temp":28.600000381469727}
bridge: Received from 539549857 msg={"id":8,"hum":77.0999984741211,"temp":28.799999237060547}
bridge: Received from 2829300856 msg={"id":9,"hum":76.5999984741211,"temp":28.600000381469727}
bridge: Received from 2829692915 msg={"id":7,"hum":76.699996948242188,"temp":28.600000381469727}
bridge: Received from 539549857 msg={"id":8,"hum":77.0999984741211,"temp":28.799999237060547}
bridge: Received from 2829300856 msg={"id":9,"hum":76.699996948242188,"temp":28.600000381469727}
bridge: Received from 2829692915 msg={"id":7,"hum":76.800003051757812,"temp":28.600000381469727}
bridge: Received from 539549857 msg={"id":8,"hum":77.0999984741211,"temp":28.799999237060547}
bridge: Received from 2829300856 msg={"id":9,"hum":76.699996948242188,"temp":28.600000381469727}
```

Gambar 4.7 Proses penerimaan data sensor 3 Node ESP-DHT22

Pada Gambar 4.7 sama halnya dengan Gambar 4.5 dan Gambar 4.6, pada Gambar 4.7 menunjukkan hasil penerimaan data sensor antara masing-masing *mesh* yang dimana ketiga *node* yang ada saling terkoneksi satu sama lain yang ditandai dengan hasil *received* antara node *id* dari masing-masing node yang dimiliki masing-masing *node.* Penerapan hasil jangkauan node mesh untuk ESP-DHT22 menerima hasil dari ketiga node berdasarkan pada perhitungan penerimaan pemantauan Suhu dan Kelembapan Lingkungan

yaitu *id* berisikan node number dan isi dari penginisialisasian *temp* yaitu *Temperature* dan *hum* yaitu *Humidity*.

Pengujian Terdistribusi

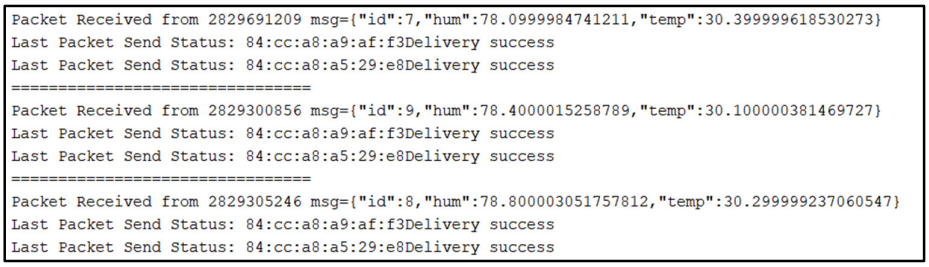
Pengujian Terdistribusi dilakukan dengan menerapkan 3 master dalam pengujian Skenario I atau yang ada pada Gambar 3.1. Master ini akan memperoleh data dari masing-masing pengelompokan mesh lalu disimpan ke masing-masing master, dan master akan melakukan pendistribusian data yang telah diperoleh sebelumnya dengan mengaplikasikan Protokol *ESP-NOW* sebagai pendistribusiannya, Gambar 4.8, Gambar 4.9, dan Gambar 4.10 Menunjukkan hasil pendistribusian yang telah dilakukan oleh masing masing *Master.* Dan hasilnya dijabarkan dalam Tabel 4.3

```
Packet Received from 539550132 msg={"id":1,"lux":65535}
Last Packet Send Status: 84:cc:a8:a5:29:e8Delivery success
Last Packet Send Status: 84:cc:a8:a8:fb:09Delivery success
================================
Packet Received from 539157004 msg={"id":3,"lux":3}
Last Packet Send Status: 84:cc:a8:a5:29:e8Delivery success
Last Packet Send Status: 84:cc:a8:a8:fb:09Delivery success
================================
Packet Received from 539538257 msg={"id":2,"lux":2}
Last Packet Send Status: 84:cc:a8:a5:29:e8Delivery success
Last Packet Send Status: 84:cc:a8:a8:fb:09Delivery success
================================
```

Gambar 4.8 Proses Pengiriman/Pendistribusian Mesh Oleh Master 1

```
Packet Received from 2829316923 msg={"id":4,"ppm":7378.78076171875}
Last Packet Send Status: 84:cc:a8:a9:af:f3Delivery success
Last Packet Send Status: 84:cc:a8:a8:fb:09Delivery success
================================
Packet Received from 2829369860 msg={"id":5,"ppm":312673.4375}
Last Packet Send Status: 84:cc:a8:a9:af:f3Delivery success
Last Packet Send Status: 84:cc:a8:a8:fb:09Delivery success
================================
Packet Received from 2829328707 msg={"id":6,"ppm":19787.083984375}
Last Packet Send Status: 84:cc:a8:a9:af:f3Delivery success
Last Packet Send Status: 84:cc:a8:a8:fb:09Delivery success
================================
```

Gambar 4.9 Proses Pengiriman/Pendistribusian Mesh Oleh Master 2



Gambar 4.10 Proses Pengiriman/Pendistribusian Mesh Oleh Master 3

Masing-masing master telah terdistribusi ke sesama master lain lewat *Mesh* 1, 2 dan 3. Setiap *master* telah menerima hasil *mesh* yang telah dikirim sebelumnya, selanjutnya dilakukan pendistribusian master ke master lain yang dapat dilihat pada Gambar 4.8, Gambar 4.9 dan Gambar 4.10. Setiap Gambar menjelaskan masing-masing proses pendistribusian *mesh* ke master yang ditandai dengan *send paket* data sensor yang mengambil MAC Address dari masing-masing *Master*

Tabel 4.1 Hasil Pendistribusian

MESH	MAC ADDRESS	NODE	BROADCAST MESSAGE	KETERANGAN
1	84:CC:A8:A9:AF:F3	1	Distribute To Mesh Master 2 & 3	OK
		2	Distribute To Mesh Master 2 & 3	OK
		3	Distribute To Mesh Master 2 & 3	OK
2	84:CC:A8:A5:29:E8	4	Distribute To Mesh Master 1 & 3	OK
		5	Distribute To Mesh Master 1 & 3	OK
		6	Distribute To Mesh Master 1 & 3	OK
3	84:CC:A8:A8:FB:09	7	Distribute To Mesh Master 1 & 2	OK
		8	Distribute To Mesh Master 1 & 2	OK
		9	Distribute To Mesh Master 1 & 2	OK

Tabel 4.1 menjelaskan hasil keseluruhan distribusi yang dijabarkan dalam bentuk tabel. Proses pendistribusian antar *master* ini dilakukan dengan mengaplikasikan pengiriman berdasarkan MAC Address masing-masing *Master* dan juga berisikan broadcast message atau status pengiriman ke master lain, ketika distribusi berhasil

maka paket yang berisikan data sensor masing-masing mesh yang telah diterima sebelumnya akan melakukan distribusi sesuai rute yang diberikan

Tabel 4.2 Hasil data pengukuran masing-masing node keseluruhan skenario 1

MESH	NODE	STATUS NODE		DATA NODE MENJADI MESH			DATA NODE MESH DITERIMA MASTER			PENDISTRIBUSIAN MASING-MASING PERANGKAT MASTER			DITERIMA SERVER		KETERANGAN
		KET	TRANSFER KE-NODE LAIN	ESP-TSL2561	ESP-MQ135	ESP-DHT22	MASTER 1	MASTER 2	MASTER 3	MASTER 1	MASTER 2	MASTER 3	MASUK	TIDAK	
1	1	*Aktif*	-	✓			✓			✓	✓	✓	✓		OK
	2	*Aktif*	-	✓			✓			✓	✓	✓	✓		OK
	3	*Aktif*	-	✓			✓			✓	✓	✓	✓		OK
2	4	*Aktif*	-		✓			✓		✓	✓	✓	✓		OK
	5	*Aktif*	-		✓			✓		✓	✓	✓	✓		OK
	6	*Aktif*	-		✓			✓		✓	✓	✓	✓		OK
3	7	*Aktif*	-			✓			✓	✓	✓	✓	✓		OK
	8	*Aktif*	-			✓			✓	✓	✓	✓	✓		OK
	9	*Aktif*	-			✓			✓	✓	✓	✓	✓		OK

Penjabaran dari Tabel 4.2 berisikan keseluruhan proses data pengukuran yang ada pada skenario 1 dan berdasarkan masing-masing gambar pengukuran proses penerimaan data mesh terlebih dahulu diberikan id node agar mengetahui identitas dari perangkat, status node difungsikan sebagai aktif dan tidaknya suatu node tersebut untuk dapat mengirimkan paket ke mesh masing-masing didalam tabel 4.1 terlihat semua pengujian yang dilakukan berhasil mentransfer node ke masing-masing mesh dan mengirimkan paket data sensor ke masing-masing master dan mampu melakukan distribusi antar master walaupun node dari perangkat mesh dalam keadaan aman, data yang telah didistribusi ini lalu dikirimkan me server dengan melakukan proses komunikasi MQTT publish master lalu ke MQTT subscribe server dengan baik, ini membuktikan dari pengiriman data pengukuran untuk skenario 1 mampu melakukan pengimplementasian mesh hingga pendistribusian perangkat oleh master mampu dijalankan secara bersamaan antar masing-masing mesh

b) Pengujian dengan dua node transmitter dalam keadaan off

Gambar 4.11 Pengujian dua node transmitter dalam keadaan off

Dalam pengujian node off pada Gambar 4.11 perangkat yang digunakan sama dengan pengujian node keseluruhan dibutuhkan masing-masing 3 node dari masing-masing pengukuran pemantauan lingkungan namun node off diinisialisasikan dengan salah satu node router down, secara otomatis pengujian yang akan dilakukan node akan dilewatkan melalui jalur yang berbeda sehingga dalam satu mesh tersebut akan tetap melakukan pengiriman data pada sensor node lain untuk dapat terhubung ke satu sama lain dan dapat saling berkomunikasi diantara node-node dalam tiap-tiap mesh tersebut dan mendistribusikan data dari *mesh* agar dapat diteruskan ke *server*.

```
bridge: Received from 539157004 msg={"id":3,"lux":2}
bridge: Received from 539538257 msg={"id":2,"lux":3}
bridge: Received from 539550132 msg={"id":1,"lux":6}
bridge: Received from 539157004 msg={"id":3,"lux":3}
bridge: Received from 539538257 msg={"id":2,"lux":3}
bridge: Received from 539550132 msg={"id":1,"lux":6}
bridge: Received from 539157004 msg={"id":3,"lux":4}
bridge: Received from 539550132 msg={"id":1,"lux":6}
bridge: Received from 539538257 msg={"id":2,"lux":3}
bridge: Received from 539157004 msg={"id":3,"lux":4}
```

Gambar 4.12 Proses pengiriman dan penerimaan data sensor apabila 2 node mati

Adapun proses penerimaan data sensor pada *mesh* dapat dilihat pada Gambar 4.12 dan penjelasan detailnya pada Tabel 4.3

Tabel 4.3 Hasil data pengukuran apabila node transmitter off

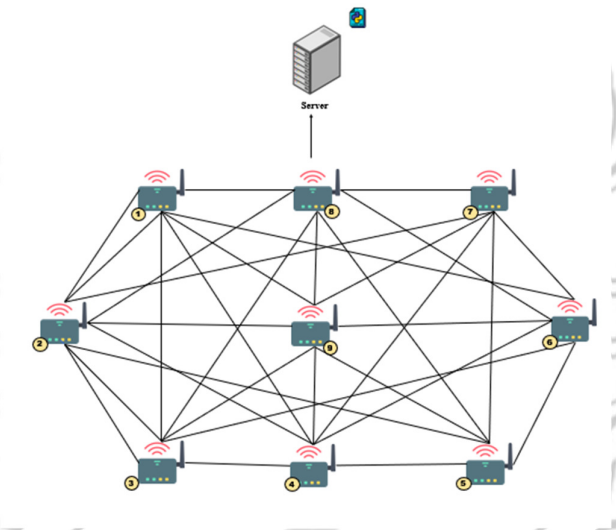
MESH	NODE	STATUS NODE		DATA NODE MENJADI MESH			DATA NODE MESH DITERIMA MASTER			PENDISTRIBUSIAN MASING-MASING PERANGKAT MASTER			DITERIMA SERVER		KETERANGAN
		KET	TRANSFER KE-NODE LAIN	ESP-TSL2561	ESP-MQ135	ESP-DHT22	MASTER 1	MASTER 2	MASTER 3	MASTER 1	MASTER 2	MASTER 3	MASUK	TIDAK	
1	1	*Aktif*	-	√			√			√	√	√	√		OK
	2	*Aktif*	-	√			√			√	√	√	√		OK
	3	*Mati*	*1 dan 2*	√			√			√	√	√	√		OK
2	4	*Aktif*	-		√			√		√	√	√	√		OK
	5	*Mati*	*4 dan 6*		√			√		√	√	√	√		OK
	6	*Aktif*	-		√			√		√	√	√	√		OK
3	7	*Aktif*	-			√			√	√	√	√	√		OK
	8	*Aktif*	-			√			√	√	√	√	√		OK
	9	*Aktif*	-			√			√	√	√	√	√		OK

Pada Tabel 4.3 pengujian data pengukuran apabila node transmitter off yang sebagai pengukuran data keseluruhan proses pengimplementasian skenario 1 apabila ada node dalam keadaan mati, terlihat pada Tabel 4.2 dilakukan pengujian 2 node dalam keadaan mati di lokasi *mesh* yang berbeda, dalam kasus ini agar data yang ada pada perangkat node ini mampu mengirimkan datanya dilakukan transfer node atau mengubah jalur rute sebelumnya menjadi ke sesama meshnya atau node yang aktif. Ini dilakukan agar node yang tadinya mati tersebut supaya dapat tetap mengirimkan paket walaupun node identitasnya berstatus mati yang dapat dilihat prosesnya pada Gambar 4.12. Sehingga ketika sampai di tahap Penerimaan data Master proses data aman dan mampu dilanjutkan ke pendistribusian agar data node yang tadinya dilakukan ke rute lain dilanjutkan lagi pendistribusiannya di master lain agar data sampai hingga ke tahap penerimaan server.

4.2.2 Pengujian Penerimaan Data Skenario 2

a) Pengujian keseluruhan node transmitter

Pengujian keseluruhan node transmitter dilakukan pengiriman antar masing-masing perangkat node ke node lain dan dipadukan dengan sensor data yang berbeda-beda guna melihat data sensor yang bervariasi yang dimiliki masing-masing node sensor ini dikirim ke board masing-masing secara *continue* terus menerus menggunakan protocol *painlessmesh* seperti yang digunakan sebelumnya pada pengujian skenario satu.



Gambar 4.13 Pengujian keseluruhan node transmitter

Untuk pengujian skenario 2 pada Gambar 4.13 memerlukan 9 node atau jumlah total dari keseluruhan node mesh yang dimiliki, keseluruhan dari 9 node ini adalah jumlah dari masing-masing dari 3 pengukuran yang dibagi menjadi 3 node yang diperlukan pada pengujian skenario 1. Pengujian ini dilakukan dilakukan secara bersamaan untuk melihat bagaimana pengiriman dan penerimaan data masing-masing node mesh apabila saling terhubung satu salam lain dalam

mengirimkan data yang memiliki konfigurasi perangkat yang berbeda-beda. Adapun proses penerimaan dan pengiriman data ke masing-masing node secara bersamaan

```
bridge: Received from 539538257 msg={"id":2,"lux":1}
bridge: Received from 2829316923 msg={"id":4,"ppm":4739.99169921875}
bridge: Received from 2829369860 msg={"id":5,"ppm":30812.470703125}
bridge: Received from 2829328707 msg={"id":6,"ppm":14979.3310546875}
bridge: Received from 539157004 msg={"id":3,"lux":1}
bridge: Received from 539550132 msg={"id":1,"lux":1}
bridge: Received from 2829300856 msg={"id":9,"hum":76.5,"temp":29}
bridge: Received from 2829692915 msg={"id":7,"hum":77.5999984741211,"temp":28.899999618530273}
bridge: Received from 539549857 msg={"id":8,"hum":77.5999984741211,"temp":28.899999618530273}
bridge: Received from 2829369860 msg={"id":5,"ppm":30812.470703125}
bridge: Received from 539538257 msg={"id":2,"lux":1}
bridge: Received from 2829316923 msg={"id":4,"ppm":4707.3662109375}
bridge: Received from 2829328707 msg={"id":6,"ppm":14979.3310546875}
bridge: Received from 539157004 msg={"id":3,"lux":1}
bridge: Received from 539550132 msg={"id":1,"lux":0}
bridge: Received from 2829369860 msg={"id":5,"ppm":30939.482421875}
bridge: Received from 2829300856 msg={"id":9,"hum":76.5,"temp":29}
bridge: Received from 2829692915 msg={"id":7,"hum":77.5999984741211,"temp":28.899999618530273}
bridge: Received from 539549857 msg={"id":8,"hum":77.699996948242188,"temp":28.899999618530273}
```

Gambar 4.14 Proses penerimaan keseluruhan data sensor secara bersamaan

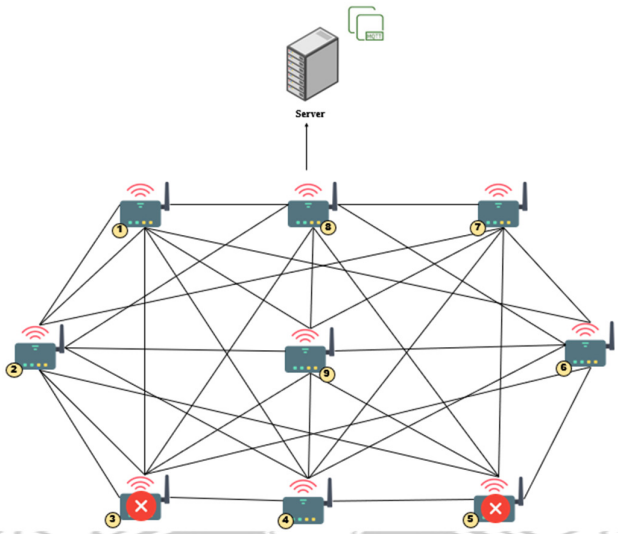
Gambar 4.14 menunjukkan hasil pengiriman dan penerimaan keseluruhan data sensor yang diuji secara bersamaan, dilihat pada Gambar 4.14 juga menjabarkan beberapa node yang berhasil ditangkap sebagai hasil distribusi akhir yang telah dilakukan masing-masing node dalam mengimplementasikan jaringan mesh dengan bantuan *painlessmesh* agar mampu mengirim data sensor ke board lain dan saling menerima satu sama lain walaupun memiliki data yang variatif. Dalam gambar juga memperlihatkan bahwa node satu sama lain dapat terbaca dan menerima hasil nya dijabarkan dalam Tabel 4.4

Tabel 4.4 Data Pengukuran Masing-Masing Node Keseluruhan Skenario 2

NODE	STATUS NODE		DATA NODE MENJADI MESH									DATA YANG DITERIMA SERVER		KETERANGAN
	KET	TRANSFER KE-NODE LAIN	1	2	3	4	5	6	7	8	9	MASUK	TIDAK	
1	*Aktif*	-	✓									✓		OK
2	*Aktif*	-		✓								✓		OK
3	*Aktif*	-			✓							✓		OK
4	*Aktif*	-				✓						✓		OK
5	*Aktif*	-					✓					✓		OK
6	*Aktif*	-						✓				✓		OK
7	*Aktif*	-							✓			✓		OK
8	*Aktif*	-								✓		✓		OK
9	*Aktif*	-									✓	✓		OK

b) Pengujian dengan dua node transmitter dalam keadaan off

Penggunaan uji node off ini menguji perangkat yang ada pada perancangan skenario dua, pengujian node dalam keadaan off ini telah dilakukan sebelumnya pada skenario pengujian satu namun untuk penerapan dalam pengujian skenario dua diinisialisasikan dengan salah satu node router down, secara otomatis pengujian yang akan dilakukan node akan dilewatkan melalui jalur yang berbeda sehingga dalam satu skenario tersebut akan tetap melakukan pengiriman data pada node lain.



Gambar 4.15 Pengujian dua node transmitter dalam keadaan off

Untuk pengujian skenario 2 pada Gambar 4.15 memerlukan 9 node atau jumlah total dari keseluruhan node mesh yang dimiliki, keseluruhan dari 9 node ini. Pengujian ini dilakukan dilakukan secara bersamaan dan melihat bagaimana dan apa yang terjadi ketika node dalam keadaan mati melakukan pengiriman dan penerimaan data masing-masing node mesh apabila saling terhubung satu sama lain dalam mengirimkan data yang memiliki konfigurasi perangkat yang berbeda-beda.

```
bridge: Received from 539157004 msg={"id":3,"lux":0}
bridge: Received from 539538257 msg={"id":2,"lux":0}
bridge: Received from 2829316923 msg={"id":4,"ppm":11277.6943359375}
bridge: Received from 2829692915 msg={"id":7,"hum":77.800003051757812,"temp":28.799999237060547}
bridge: Received from 2829300856 msg={"id":9,"hum":76.199996948242188,"temp":29.100000381469727}
bridge: Received from 539549857 msg={"id":8,"hum":78.800003051757812,"temp":28.5}
bridge: Received from 2829328707 msg={"id":6,"ppm":13475.732421875}
bridge: Received from 539157004 msg={"id":3,"lux":0}
bridge: Received from 539538257 msg={"id":2,"lux":1}
bridge: Received from 2829316923 msg={"id":4,"ppm":11277.6943359375}
bridge: Received from 2829692915 msg={"id":7,"hum":77.800003051757812,"temp":28.799999237060547}
bridge: Received from 2829300856 msg={"id":9,"hum":76.199996948242188,"temp":29.100000381469727}
bridge: Received from 539549857 msg={"id":8,"hum":78.800003051757812,"temp":28.5}
bridge: Received from 2829328707 msg={"id":6,"ppm":13475.732421875}
bridge: Received from 539157004 msg={"id":3,"lux":0}
```

Gambar 4.16 Proses pengiriman dan penerimaan data sensor node off

Pada Gambar 4.16 Proses yang terjadi ialah node 1 dan node 5 tidak terbaca oleh salah satu node yang terhubung namun disisi lain apabila dilakukan pengecekan pada node lain yang terlihat pada Gambar 4.16 node id yang mati tersebut tidak melakukan pengiriman data namun tidak memberhentikan node yang lain untuk melakukan pengiriman ke node lain membuktikan bahwa node yang mati tadinya dapat mampu mendistribusi data melalui jalur lain apabila node down dan segera melakukan pengiriman lain ke jalur yang dirasa mampu menerima ke node lainnya

Tabel 4.5 Hasil data pengukuran dua node transmitter dalam keadaan off

NODE	STATUS NODE		DATA NODE MENJADI MESH									DATA YANG DITERIMA SERVER		KETERANGAN
	KET	TRANSFER KE-NODE LAIN	1	2	3	4	5	6	7	8	9	MASUK	TIDAK	
1	*Aktif*	-	✓									✓		OK
2	*Aktif*	-		✓								✓		OK
3	*Mati*	*Remote*			X							✓		OK
4	*Aktif*	-				✓						✓		OK
5	*Mati*	*Remote*					X					✓		OK
6	*Aktif*	-						✓				✓		OK
7	*Aktif*	-							✓			✓		OK
8	*Aktif*	-								✓		✓		OK
9	*Aktif*	-									✓	✓		OK

Sama halnya pada tabel 4.4 pengujian data pengukuran apabila node mati untuk skenario 2 ini memiliki pengukuran serupa pengukuran data keseluruhan proses pengimplementasian skenario 2 ini apabila ada node dalam keadaan mati, terlihat pada tabel 4.5 ini dilakukan pengujian 2 node dalam keadaan mati di lokasi *mesh* dan node yang serupa dengan skenario 1, namun dalam kasus skenario 2 ini pengaplikasan lewat rute lain cukup berbeda untuk skenario 2 ini mampu melakukan transfer node tidak hanya dalam lingkup mesh nya karna untuk skenario 2 ini mesh yang digunakan hanya 1 dan tanpa menggunakan master sehingga tidak ada pendistribusian master melainkan pendistribusian hanya

dilakukan oleh setiap node dan mengirimkan langsung datanya ke server. untuk node yang mati tadi. skenario 2 rute yang dilalui hanya mampu menggunakan transfer node ke node terdekatnya untuk mengirimkan data sensornya sehingga node yang tadinya mati tersebut dapat tetap mengirimkan paket walaupun node identitasnya berstatus mati yang dapat dilihat prosesnya pada Gambar 4.15 dan identitas node yang mati tadi tadi dapat diliat di node terdekatnya seperti yang tertera. Sehingga ketika sampai di tahap masing-masing menjadi mesh proses data aman dan mampu dilanjutkan ke pengiriman server

4.2.5 Node ke *Server* Skenario 1 & 2

Data sensor yang telah diterima oleh baik dalam pengujian skenario I dan skenario 2 selanjutnya akan diliat hasil pengiriman dan proses *publish* ke protokol MQTT. Penerapan hasil jangkauan node yang telah dijabarkan pada Bab 3 pengujian dilakukan baik skenario satu dan skenario dua. Skenario 1 diperlukan beberapa konfigurasi program didalam *master* dan diperlukan pengujian secara bersaamaan agar mengetahui data yang dikirim saling mengirim dan menerima antar masing-masing esp-mesh namun untuk itu terlebih dahulu master memerlukan *message*/*topic* tertentu sebagai transaksi data dari node-node ke MQTT, Skenario 2 diperlukan Topic pesan/subscribe yang dirancang pada masing-masing node untuk penerimaan data sensor dijabarkan pada Gambar 4.17, Gambar 4.18, Gambar 4.19. Topik subscribe dirancang sesuai dengan topik yang dikirim oleh publisher agar data yang dikirim dapat diterima dan disimpan ke server

```
def on_connect(client, userdata, flags, rc):
    print("Connected.. "+str(rc))
    client.subscribe("ALLMESH")
```

Gambar 4.17 Inisialisasi Client Subscribe

```
void receivedCallback(uint32_t from, String &msg)
{
  Serial.println("================================");
  Serial.printf("Packet Received from %u msg=%s\n", from, msg.c_str());
  esp_now_send(broadcastAddress1, (uint8_t *) msg.c_str(), sizeof(msg));
  esp_now_send(broadcastAddress2, (uint8_t *) msg.c_str(), sizeof(msg));
  esp_now_send(broadcastAddress3, (uint8_t *) msg.c_str(), sizeof(msg));
  mqttClient.publish("ALLMESH", msg.c_str());
}
```

Gambar 4.18 Inisialisasi *Publish MQTT Broker* Skenario 1

```
void receivedCallback( const uint32_t &from, const String &msg ) {
  Serial.printf("bridge: Received from %u msg=%s\n", from, msg.c_str());
  String topic = "MyTopic";
  mqttClient.publish(topic.c_str(), msg.c_str());
}
```

Gambar 4.19 Inisialisasi *Publish MQTT Broker* Skenario 2

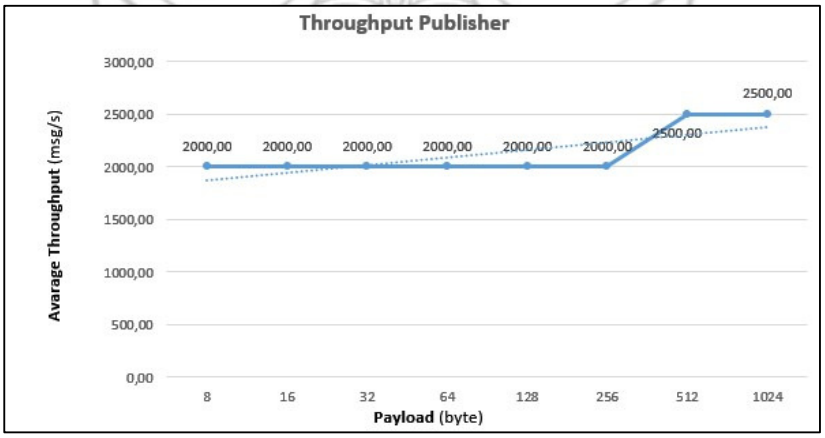
Data sensor Gambar 4.17 yang diperoleh dari pengiriman *node* diterima *master.* Gambar 4.18 dan Gambar 4.19 untuk masing-masing node di skenario 2 lalu di *subscribe* ke *server* secara terus menerus menggunakan protocol MQTT dengan bantuan broker *mosquito.* Adapun proses hasil penerimaan data pada antara masing-masing board mesh pada *server* dapat dilihat pada lampiran 4 yang berisikan hasil distribusi *master* ke broker untuk skenario 1 dan node ke broker untuk skenario 2 yang didistribusikan dengan format data JSON

4.3 Hasil Pengujian *Throughput*

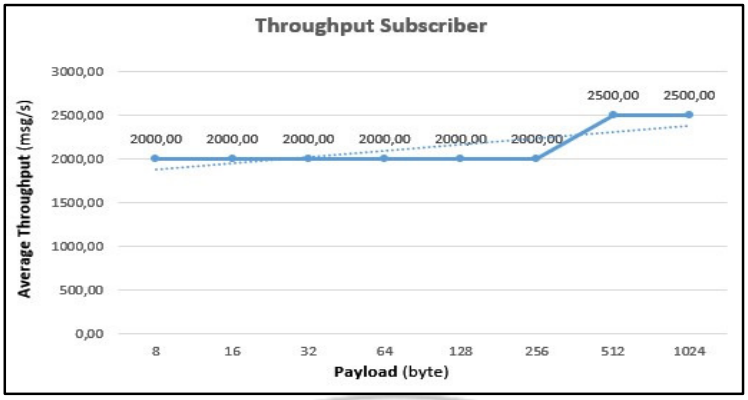
Pada pengujian ini bertujuan untuk mengetahui akan kemampuan yang dimiliki oleh masing-masing *node* ESP-MESH pada setiap kapasitas jaringan yang digunakan dan berapa rentang waktu yang dibutuhkan dalam mentransmisi *message* dalam satuan waktu. Untuk penerimaan data *master* ke *server* sebelumnya telah berhasil penerimaan data dari node *mesh* masing-masing ke dalam *master* sebagai wadah penyimpanan data sensor yang dilakukan hasil *throughput* dengan mengirimkan jumlah dari *record* sebesar 10.000. Size *record* yang diterapkan dalam penelitian ini mengambil rentang waktu perbyte yaitu 8, 16, 32, 64, 128, 256, 512 dan 1024 *bytes.*

A) Pengujian Throughput *Publisher* dan *Subscribe* ESP-TSL2561

Hasil Pengujian *throughput* dengan 3 *node* dengan menggunakan variasi record size sebanyak 8, 16, 32, 64, 128, 256, 512 dan 1024 dapat dilihat pada Gambar 4.20 dan Gambar 4.21 menunjukkan bahwa pada grafik *throughput publisher* dan *subscriber*.



Gambar 4.20 Grafik *throughput* publisher 3 Node ESP-TSL2561



Gambar 4.21 Grafik *throughput* subscribe 3 Node ESP-TSL2561

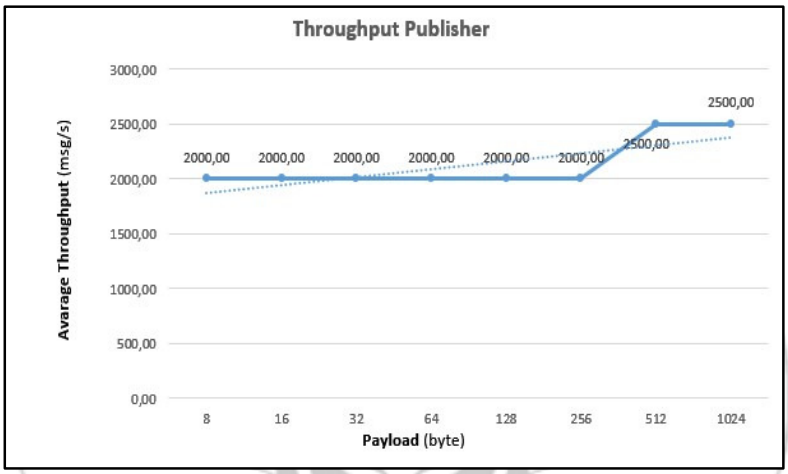
Pada Gambar 4.20 dapat dilihat jumlah *throughput publish* dengan 3 *node* menunjukkan bahwa pada grafik *throughput* tidak menampilkan ada pergerakan ukuran pesan yang ada pada *size* / *record* di 8 byte hingga 128 byte, namun di bytes 216 memiliki peningkatan *throughput* menjadi 2500,00 dan seimbang setelah melewati 216 byte di 512 *byte* hingga *bytes* 1024. *Throughput* yang dihasilkan oleh data ESP-TSL2561 berada pada titik 2500,00. Lalu untuk *throughput subscribe* untuk Gambar 4.21 *throughput subscribe* tidak menampilkan data dan pergerakan ukuran size yang dihasilkan namun di 216 byte – 512 *byte* hingga 1024 *bytes*. Untuk lebih detail dari grafik yang diperoleh dijabarkan lebih lanjut pada Tabel 4.6 Dengan jumlah message sebanyak 10000 data Dimana setiap *messagenya* berukuran 8, 16, 32, 64, 128, 256, 512 dan 1024 *bytes.*

Tabel 4.6 Nilai Throughput ESP-TSL2561

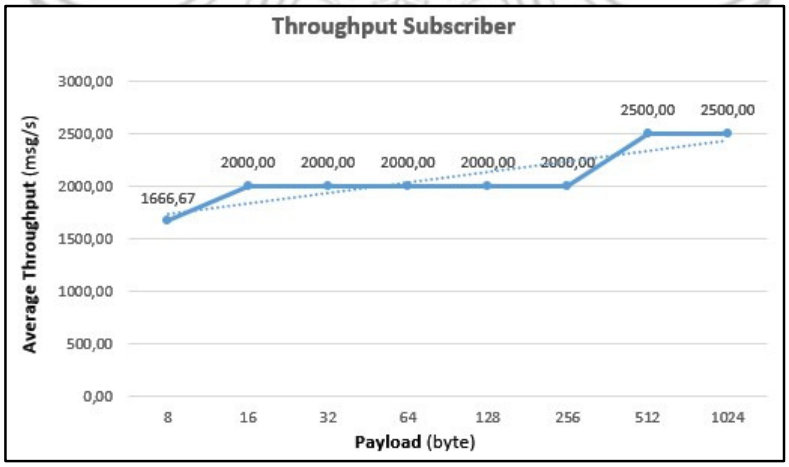
Payload (byte)	Average Throughput Publisher	Average Throughput Subscriber
8	2000,00	2000,00
16	2000,00	2000,00
32	2000,00	2000,00
64	2000,00	2000,00
128	2000,00	2000,00
256	2000,00	2000,00
512	2500,00	2500,00
1024	2500,00	2500,00

A) Pengujian Throughput *Publisher* dan *Subscribe 3 Node* ESP-MQ135

Hasil Pengujian *throughput* dengan 3 *node* dengan menggunakan variasi record size sebanyak 8, 16, 32, 64, 128, 256, 512 dan 1024 dapat dilihat pada Gambar 4.22 dan 4.23 menunjukkan bahwa pada grafik *throughput publisher* dan *subscribe* ESP MQ-135



Gambar 4.22 Grafik *throughput* publisher 3 Node ESP-MQ135



Gambar 4.23 Grafik *throughput* subscribe 3 Node ESP-MQ135

Pada Gambar 4.22 dapat dilihat jumlah *throughput publish* dengan 3 *node* untuk ESP-TSL2561 menunjukkan bahwa pada grafik *throughput* tidak menampilkan ada

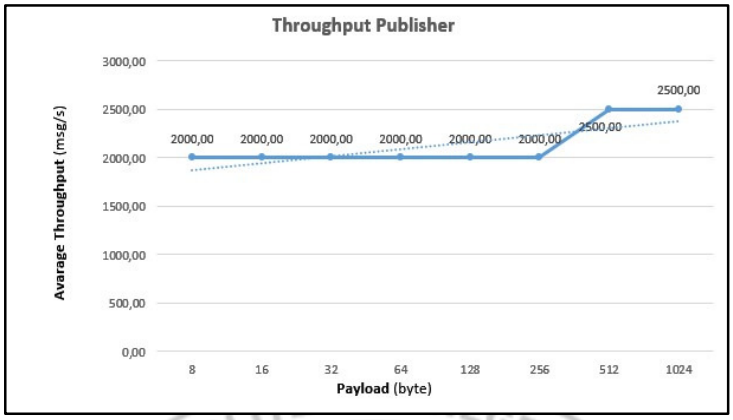
pergerakan ukuran pesan yang ada pada *size* / *record* di 8 byte pertama hingga 128 byte, namun di bytes 216 memiliki peningkatan menjadi 2500,00 dan seimbang setelah melewati 216 byte di 512 hingga 1024 *throughput* yang dihasilkan oleh data ESP-TSL2561 berada pada titik 2500,00. Lalu untuk *throughput subscribe* Gambar 4.23 Untuk *throughput publish* di 8 *byte* pertama data dimulai pada 1666,67 dan di size 16 memulai pergerakan ukuran pesan menjadi 200,00 dan berjalan stabil hingga di size 256 byte dan meningkat kembali pada byte 512 menjadi 2500,00 dan stabil hingga di byte 1024. Untuk lebih detail dari grafik yang diperoleh dijabarkan lebih lanjut pada Tabel 4.7 Dengan jumlah message sebanyak 10000 data Dimana setiap *messagenya* berukuran 8, 16, 32, 64, 128, 256, 512 dan 1024 *bytes.*

Tabel 4.7 Nilai Throughput ESP-MQ135

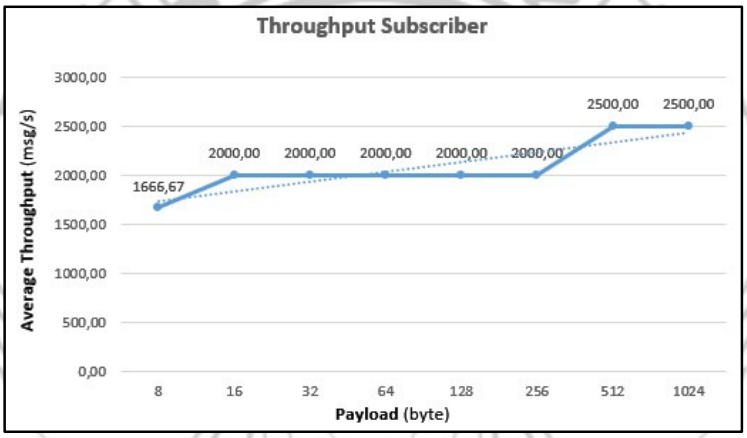
Payload (byte)	Average Throughput Publisher	Average Throughput Subscriber
8	2000,00	1666,67
16	2000,00	2000,00
32	2000,00	2000,00
64	2000,00	2000,00
128	2000,00	2000,00
256	2000,00	2000,00
512	2500,00	2500,00
1024	2500,00	2500,00

B) Pengujian Throughput *Publisher* dan *Subscribe* 3 Node ESP-DHT22

Hasil Pengujian *throughput* dengan 3 *node* dengan menggunakan variasi record size sebanyak 8, 16, 32, 64, 128, 256, 512 dan 1024 *bytes* dapat dilihat pada Gambar 4.24 dan Gambar 4.25 menunjukkan bahwa pada grafik *throughput publisher* dan *subscribe* ESP-DHT22



Gambar 4.24 Grafik *throughput* publisher ESP-DHT22



Gambar 4.25 Grafik *throughput subscribe* ESP-DHT22

Pada Gambar 4.24 dapat dilihat jumlah *throughput publish* dengan 3 *node* untuk ESP-DHT22 sama halnya pada *node* sebelumnya yaitu pada ESP-MQ135 menunjukkan bahwa pada grafik *throughput* tidak menampilkan ada pergerakan ukuran pesan yang ada pada *size* / *record* di 8 byte pertama hingga 256 byte, namun di bytes 512 memiliki peningkatan menjadi 2500,00 dan seimbang setelah melewati 216 byte di 512 hingga *byte* di 1024 *throughput* yang dihasilkan oleh data ESPDHT-22 berada pada titik 2500,00. Lalu untuk *throughput subscribe* sama halnya ukuran size ESP-DHT22 sebelumnya Pada Gambar 4.25 Untuk *throughput*

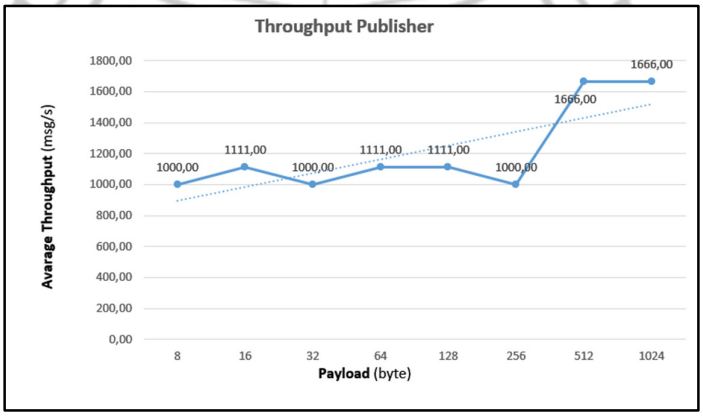
subscribe di 8 *byte* pertama data dimulai pada 1666,67 dan di size 16 memulai pergerakan ukuran pesan menjadi 200,00 dan berjalan stabil hingga di size 256 byte dan meningkat kembali pada byte 512 menjadi 2500,00 dan stabil hingga di byte 1024. Untuk lebih detail dari grafik yang diperoleh dijabarkan lebih lanjut pada Tabel 4.8 Dengan jumlah message sebanyak 10000 data

Tabel 4.8 Nilai Throughput ESP-DHT22

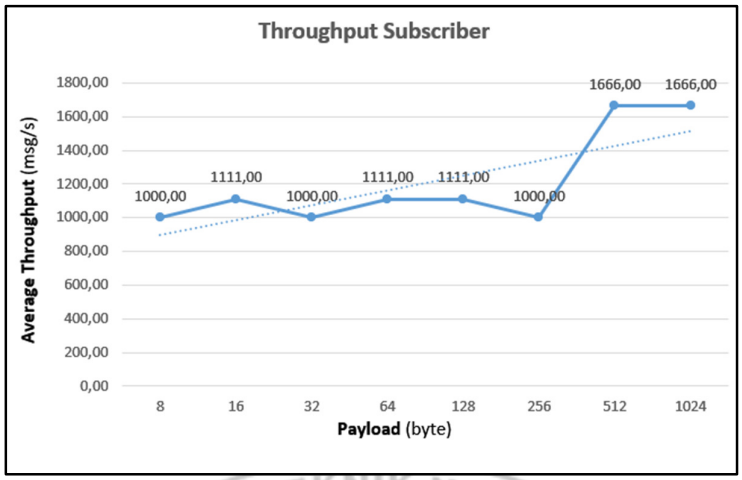
Payload (byte)	Average Throughput Publisher	Average Throughput Subscriber
8	2000,00	1666,67
16	2000,00	2000,00
32	2000,00	2000,00
64	2000,00	2000,00
128	2000,00	2000,00
256	2000,00	2000,00
512	2500,00	2500,00
1024	2500,00	2500,00

C) Pengujian Throughput *Publisher* dan *Subscribe* 9 Node Bersamaan

Hasil Pengujian *throughput* dengan 9 *node* dengan menggunakan variasi record size sebanyak 8, 16, 32, 64, 128, 256, 512 dan 1024 dapat dilihat pada Gambar 4.26 dan Gambar 4.27 menunjukkan bahwa pada grafik *throughput publisher* dan *subscribe*



Gambar 4.26 Grafik *throughput publisher* 9 Node



Gambar 4.27 Grafik *throughput subscribe* 9 Node

Gambar 4.26 dapat dilihat jumlah *throughput publish* dengan 9 *node* secara bersamaan untuk ke-3 jenis pengukuran data yang di rancang menjadi satu menunjukkan bahwa pada grafik *throughput* tidak menampilkan ada pergerakan ukuran pesan yang berbeda baik itu di througput publish dan throughput subscribe, karna keduanya memiliki average throughput yang serupa yang telah dijambarkan Pada Gambar 4.26 dan Gambar 4.27. Untuk pengukuran throughput publish dan subscribe di *size / record* 8 byte pertama memiliki average sekitar 1000.00 dan memiliki kelonjakan di byte 16 sekitar 1111.00, setelah di byte kedua berikut untuk selanjutnya average memiliki posisi yang naik turun hingga di byte 256 dan di bytes 512 memiliki peningkatan menjadi 1666,00 dan seimbang setelah melewati 512 hingga *byte* di 1024. Untuk Penjabarannya bisa diliat pada Tabel 4.9

Tabel 4.9 Nilai Throughput Node Bersamaan

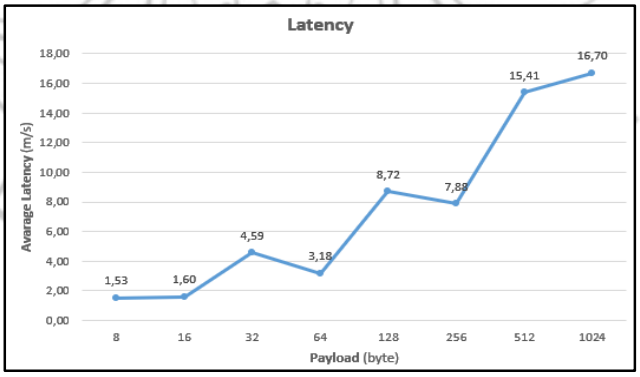
Payload (byte)	Average Throughput Publisher	Average Throughput Subscriber
8	1000,00	1000,00
16	1111,00	1111,00
32	1000,00	1000,00
64	1111,00	1111,00
128	1111,00	1111,00
256	1000,00	1000,00
512	1666,00	1666,00
1024	1666,00	1666,00

4.4 Hasil Pengujian *Latency*

Latency bertujuan untuk mengetahhui rata-rata jeda waktu pada saat pengiriman data node mesh pada saat pesan diterima oleh *master* dengan mengirimkan jumlah dari *record* sebesar 10.000. Size *record* yang diterapkan dalam penelitian ini mengambil rentang waktu perbyte yaitu 8, 16, 32, 64, 128, 256, 512 dan 1024 *bytes.*

A) Pengujian *Latency* ESP-TSL2561

Secara umum *latency* dipengaruhi oleh bertambahnya ukuran pesan. Dari grafik dapat dilihat bahwa pesan yang dikirim ke *server* semakin besar ukuran pesan yang dikirim maka waktu *latency* juga semakin besar Hasil Pengujian *latency* dengan 3 *node* untuk ESP-TSL2561 menggunakan variasi record size sebanyak 8, 16, 32, 64, 128, 256, 512 dan 1024 *bytes*. Dapat dilihat pada Gambar 4.28 menunjukkan bahwa pada grafik *latency* seiring bertambahnya ukuran pesan



Gambar 4.28 Grafik *latency* ESP-TSL2561

Berdasarkan Tabel 4.10 nilai untuk *latency* pada 8 *bytes* diawal bernilai 1,53 ms berlanjut ke *bytes* 16 menjadi 1.60 lalu mulai melonjak naik ke byte 32 menjadi 4.59 ms dan mulai mengalami penurunan di *byte* 64 menjadi 3,18 ms

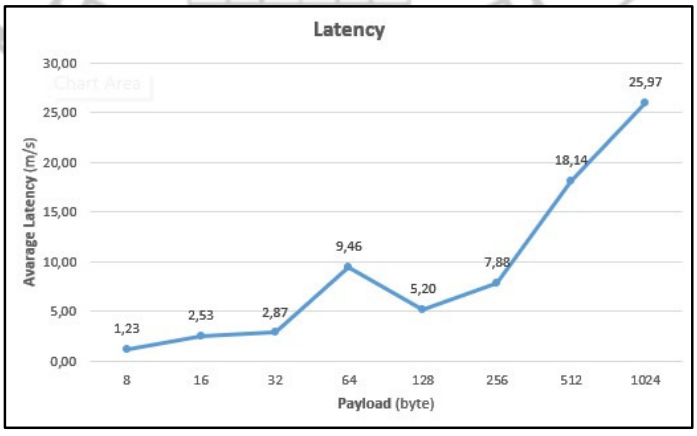
dan naik kembali di *bytes* 128 menjadi 8.72 ms dan turun kembali sekitar 0,8 ms menjadi 7,8 di 256 *bytes* dan melonjak tinggi naik 18.41 di 512 *bytes* dan naik menjadi 22,70 ms di 1024 *bytes.* Hasil grafik diperoleh dengan ditentukan jumlah *message* sebanyak 10000 data.

Tabel 4.10 Nilai *Latency* ESP-TSL2561

Size/byte	*Latency*
8	1.53 ms
16	1.60 ms
32	4.59 ms
64	3.18 ms
128	8.72 ms
256	7.88 ms
512	15.41 ms
1024	16.70 ms

B) Pengujian *Latency* ESP-MQ135

Hasil Pengujian *latency* dengan 3 *node* untuk MESH-2 dengan menggunakan variasi *record size* sebanyak 8, 16, 32, 64, 128, 256, 512 dan 1024 *bytes*. Dapat dilihat pada Gambar 4.25



Gambar 4.29 Grafik *latency* ESP-MQ135

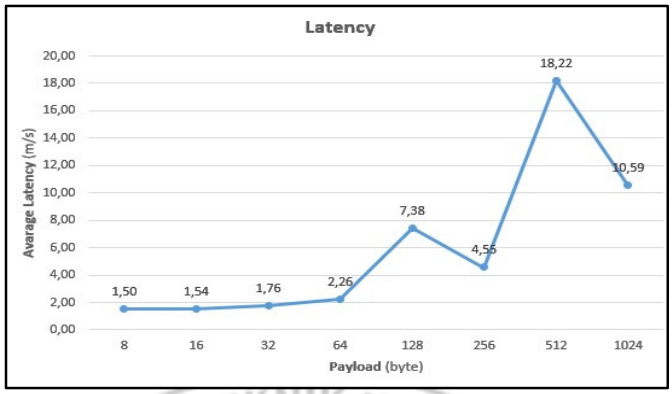
Berdasarkan Tabel 4.11 dan grafik pada Gambar 4.28 nilai untuk *latency* untuk MESH2 yang diperoleh dengan mengirimkan jumlah *message* sebanyak 10000 data. Dimana setiap *messagenya* berukuran 8, 16, 32, 64, 128, 256, 512 dan 1024 *bytes*. Pada diagram 8 *bytes* diawal nilai 1.23 ms lalu di 16 *bytes* menjadi 2.53 ms dan mulai meningkat ke 32 bytes menjadi 2.87 ms dan makin meningkat di *bytes* 64 menjadi 9.46 ms, lalu mengalami penurunan di 128 *bytes* menjadi 5.20 ms dan mulai pada 256 *bytes* melonjak naik secara terus menerus hingga 1024 ms asil grafik diperoleh dengan ditentukan jumlah *message* sebanyak 10000 data.

Tabel 4.11 Nilai *Latency* ESP-MQ135

Size/byte	*Latency*
8	1.23 ms
16	2.53 ms
32	2.87 ms
64	9.46 ms
128	5.20 ms
256	7.88 ms
512	18.14 ms
1024	25.97 ms

C) Pengujian *Latency* ESP-DHT22

Hasil Pengujian *latency* dengan 3 *node* dengan menggunakan variasi record size sebanyak 8, 16, 32, 64, 128, 256, 512 dan 1024 *bytes*. Dapat dilihat Pada Gambar 4.30



Gambar 4.30 Grafik *latency* ESP-DHT22

Berdasarkan Tabel 4.12 nilai untuk *latency* yang diperoleh dengan mengirimkan jumlah *message* sebanyak 10000 data. Di 8 Byte data terliat stabil hingga byte 64 namun ketika memasuki byte 128 mulai terjadi peningkatan *throughput* dimana sebesar 7.38 ms dan Nilai *throughput* terbesar diperoleh oleh size/record 512 *bytes.* Namun di ukuran record selanjutnya terjadi penurunan *throughput* yang dapat dilihat pada gambar 4.17 yang dimiliki oleh byte 1024. Untuk penjambaran data grafik dapat dilihat pada Tabel 4.12

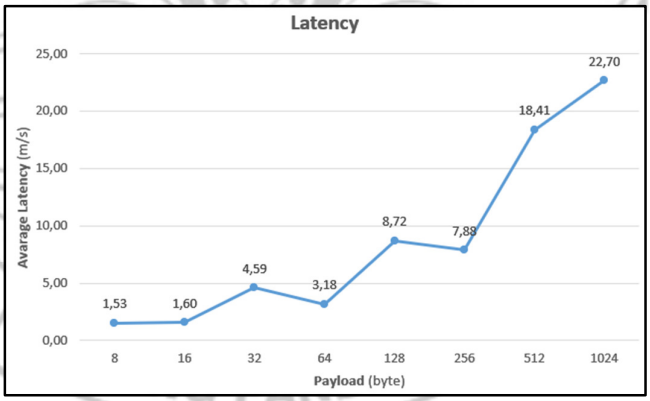
Tabel 4.12 Nilai *Latency* ESP-DHT22

Size/byte	*Latency*
8	1.50 ms
16	1.54 ms
32	1.76 ms
64	2.26 ms
128	7.38 ms

256	4.55 ms
512	18.22 ms
1024	10.55ms

D) Pengujian Throughput *Publisher* dan *Subscribe* Skenario 2

Secara umum *latency* dipengaruhi oleh bertambahnya ukuran pesan. Dari grafik dapat dilihat bahwa pesan yang dikirim ke *server* semakin besar ukuran pesan yang dikirim maka waktu *latency* juga semakin besar Hasil Pengujian *latency* dengan 9 *node* untuk semua jenis *MESH* menggunakan variasi record size sebanyak 8, 16, 32, 64, 128, 256, 512 dan 1024 *bytes*. Dapat dilihat pada Gambar 4.31 menunjukkan bahwa pada grafik *latency* seiring bertambahnya ukuran pesan



Gambar 4.31 Grafik *latency* 9 Node

Berdasarkan Tabel 4.13 nilai untuk *latency* yang diperoleh dengan mengirimkan jumlah *message* sebanyak 10000 data. Di 8 Byte data terliat stabil hingga byte 16 yang memiliki perbedaan latency sekitar di angka 1.53ms namun ketika memasuki byte 32 mulai terjadi peningkatan *latency*

dimana sebesar 4.59 ms, dari hasil pengukuran dimulai pada byte 32 hingga 1024 memiliki peningkatan dan penurunan data tidak signifikan karna seperti yang ada pada gambar terlihat lonjakan average latency yang berubah seperkian byte dan terlihat Nilai *throughput* terbesar diperoleh oleh size/record 1024 *bytes* sebesar 22.70 ms. Untuk penjambaran data grafik dapat dilihat pada Tabel 4.13

Tabel 4.13 Nilai *Latency* 9 Node

Size/byte	*Latency*
8	1.53 ms
16	1.60 ms
32	4.59 ms
64	3.18 ms
128	8.72 ms
256	7.88 ms
512	18.41 ms
1024	22.70 ms

Detail hasil pengujian throughput dan latency dengan keseluruhan node dapat dilihat dari Tabel 4.14, 4.15, 4. 16, dan 4,17

Tabel 4.14 Akumulasi throughput *publisher* dan *subscriber* Skenario I

SKENARIO I						
Payload (byte)	3 NODE ESP-TSL2561		3 NODE ESP-MQ135		3 NODE ESP-DHT22	
	Kbps	Mbps	Kbps	Mbps	Kbps	Mbps
8	2000,00	19,53	1666,00	16,27	1666,00	16,27
16	2000,00	19,53	2000,00	19,53	2000,00	19,53
32	2000,00	19,53	2000,00	19,53	2000,00	19,53
64	2000,00	19,53	2000,00	19,53	2000,00	19,53
128	2000,00	19,53	2000,00	19,53	2000,00	19,53
256	2000,00	19,53	2000,00	19,53	2000,00	19,53
512	2500,00	24,41	2500,00	24,41	2500,00	24,41
1024	2500,00	24,41	2500,00	24,41	2500,00	24,41
TOTAL	2125,00	166,00	2083,25	162,74	2083,25	162,74
AVERAGE		20,75		20,34		20,34

SKENARIO I						
Payload (byte)	3 NODE ESP-TSL2561		3 NODE ESP-MQ135		3 NODE ESP-DHT22	
	Kbps	Mbps	Kbps	Mbps	Kbps	Mbps
8	2000,00	19,53	2000,00	19,53	2000,00	19,53
16	2000,00	19,53	2000,00	19,53	2000,00	19,53
32	2000,00	19,53	2000,00	19,53	2000,00	19,53
64	2000,00	19,53	2000,00	19,53	2000,00	19,53
128	2000,00	19,53	2000,00	19,53	2000,00	19,53
256	2000,00	19,53	2000,00	19,53	2000,00	19,53
512	2500,00	24,41	2500,00	24,41	2500,00	24,41
1024	2500,00	24,41	2500,00	24,41	2500,00	24,41
TOTAL	2125,00	166,00	2125,00	166,00	2125,00	166,00
AVERAGE		20,75		20,75		20,75

Tabel 4.15 Akumulasi throughput *publisher* dan *subscriber* Skenario 2

SKENARIO II				
Payload (byte)	Average Throughput Publisher		Average Throughput Subscriber	
	Kbps	Mbps	Kbps	Mbps
8	1000,00	9,76	1000,00	9,76
16	1111,00	10,85	1111,00	10,85
32	1000,00	9,76	1000,00	9,76
64	1111,00	10,85	1111,00	10,85
128	1111,00	10,85	1111,00	10,85
256	1000,00	9,76	1000,00	9,76
512	1666,00	16,27	1666,00	16,27
1024	1666,00	16,27	1666,00	16,27
TOTAL	1208,13	94,37	1208,13	94,37
AVERAGE		11,80		11,80

Tabel 4.16 Akumulasi latency Skenario I & 2

SKENARIO 1			
Payload (byte)	3 NODE ESP-TSL2561	3 NODE ESP-MQ135	3 NODE ESP-DHT22
	Latency	Latency	Latency
8	1,53	1,23	1,50
16	1,60	2,53	1,54
32	4,59	2,87	1,76
64	3,18	2,26	2,26
128	8,72	7,38	7,38
256	7,88	4,55	4,55
512	15,41	18,22	18,22
1024	16,70	10,55	10,55
TOTAL	59,61	49,59	47,76
AVERAGE	7,45	6,20	5,97

SKENARIO 2	
Payload (byte)	Latency
8	1,53
16	1,60
32	4,59
64	3,18
128	8,72
256	7,88
512	18,41
1024	22,70
TOTAL	68,61
AVERAGE	8,58

4.5 ANALISIS

Dari hasil pengujian pada kedua Skenario, dapat disimpulkan bahwa kedua skenario pengujian yang telah dilakukan mampu melakukan pengimplementasian mesh ini dengan baik walaupun kedua skenario memiliki perbedaan dari segi pendistribusian dan pengaplikasian meshnya, dengan begitu kedua skenario memiliki keunggulan yang berbeda dan memiliki resiko pengiriman yang juga berbeda. Untuk Analisa pada kedua skenario dijabarkan dibawah ini:

4.5.1 Analisis Perangkat

1. Skenario 1

   Keunggulan

   Memiliki perangkat *master* yang membantu mendistribusikan data dari perangkat masing-masing mesh agar dapat meminimalisir node mati terdistribusi di perangkat master lain

   Kelemahan

   Perangkat *master* harus dalam keadaan on, apabila salah satu mati data node mesh pada master tersebut langsung mati dan tidak mampu melakukan pendistribusian dan pengiriman ke server

2. Skenario 2

   Keunggulan

   Perangkat meshnya mampu saling mengirim dan menerima data sensor node lain walaupun jenis sensor data yang diterima variatif

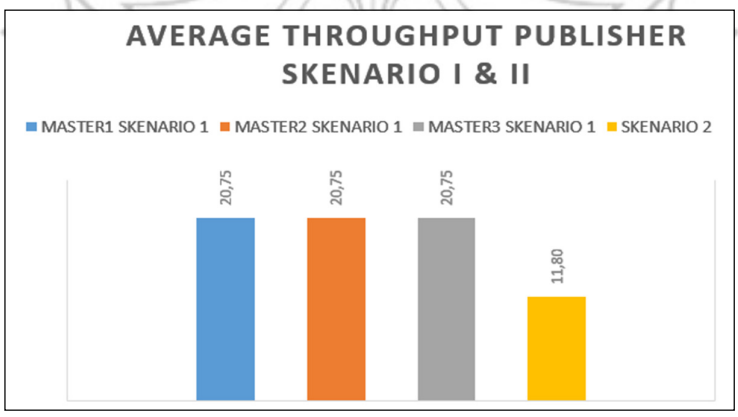
   Kelemahan

   Tidak adanya perangkat master dalam dalam pendistribusian nya sehingga pencengahan meminimalisir perangkat node mati hanya bisa diandalkan oleh node lain yang terdekat

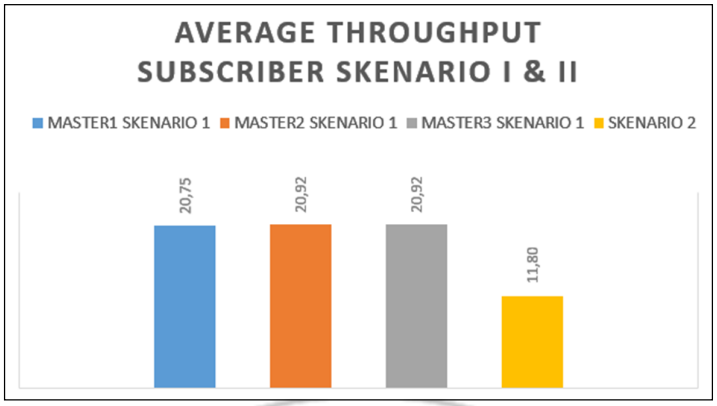
Setelah didapatkan analisa pada kedua skenario akan perangkat dari keunggulan dan kelemahannya masing-masing bisa disimpulkan bahwa kedua skenario memiliki fitur keunggulan dan kelemahannya masing-masing karna berdasarkan dari fungsi yang harus diberikan dari kedua skenario.

4.5.2 Performa Pengukuran Data (*Throughput & Latency*)

Througput dan *latency* yang telah diujikan dan dijabarkan akumulasinya pada tabel 4.14, tabel 4.15, tabel 4.16. Dapat disimpulkan bahwa kedua skenario memiliki jangkauan performa yang berbeda. Dengan menggunakan pengukuran *Throughput* yang menghitung jumlah data yang berhasil dipindahkan dari satu node ke node lain dalam periode waktu tertentu, dan biasanya diukur dalam bit per detik (bps) juga *Latency* untuk menghitung waktu yang dibutuhkan data untuk menempu jarak dari asal ke tujuan berdasarkan dari standar **Telecommunications and Internet Protocol Harmonization Over Networks** (TIPHON). Yang telah dijelaskan dalam Tabel 2.5



Gambar 4.32 Average Throughput Publisher Skenario 1 & 2

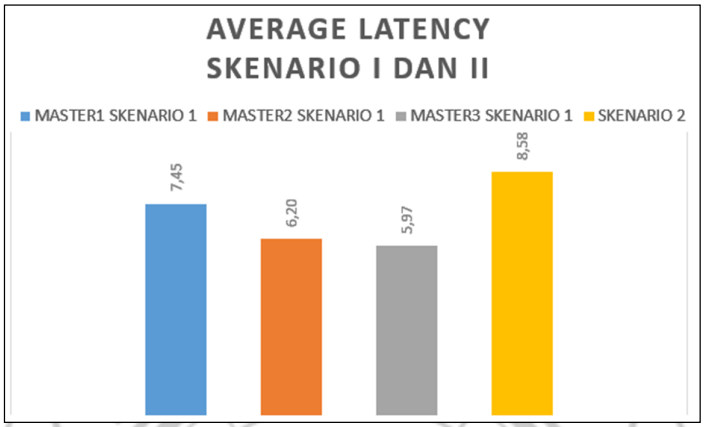


Gambar 4.33 Average Throughput Subscriber Skenario 1 & 2

Average atau rata-rata *publish* serta *subscribe* pada Gambar 4.32 dan Gambar 4.33 yang dihasilkan pada pengujian skenario 1 dan 2, maka dapat dilihat pada kedua skenario yang digunakan memiliki average publish dan subscribe pengiriman data pengukuran masing masing skenario dijabarkan perhitungan Mbps memiliki kategori *indeks* 4 atau **sangat baik** berdasarkan Tabel 4.14 dan 4.15. *Average throughput publisher & subscriber* yang dihasilkan untuk pengujian skenario 1 dan skenario 2. Skenario 1 dijumlahkan untuk setiap masternya memiliki rata-rata throughput publish 20.75 Mbps dan 20.86 Mbps untuk skenario 2 yang ada pada Gambar 4.32 dan throughput subscriber skenario 1 dan 2 sebesar 11.80 Mbps Gambar 4.33 sehingga untuk mengetahui berapa perbandingan *throughput* yang dihasilkan oleh masing-masing skenario digunakan rumus

AVERAGE THROUGHPUT PUBLISHER	**AVERAGE THROUGHPUT SUBSCRIBER**
$\boldsymbol{Perbandingan = \frac{skenario\ 1}{skenario\ 2}}$ $\boldsymbol{= \frac{20.75}{11.80} = 1,74\ Mbps}$	$Perbandingan = \frac{skenario\ 1}{skenario\ 2}$ $= \frac{20.86}{11.80} = 1,76\ Mbps$

Sehingga perbandingan *throughput publish* untuk 2 skenario sebesar 1.74 Mbps dan throughput subscriber sebesar 1.76 Mbps dalam melakukan pengiriman data



Gambar 4.34 Average Latency Skenario 1 & 2

Dengan menerapkan perhitungan dari tabel 2.6 untuk pengukuran *latency* kedua skenario yang ada pada gambar 4.34 memiliki indeks performa yang ***sangat baik*** dan berada dikategori indeks 4 perhitungan waktu yang dibutuhkan data untuk menempuh jarak ke server dimiliki masing-masing perbedaan untuk setiap master skenario 1 dan node-node untuk skenario 2. Skenario 1 dijumlahkan untuk setiap masternya memiliki rata-rata sekitar 6.54 ms *latency* yang berhasil dikeluarkan seperti yang ada pada tabel 4.16 sedangkan untuk skenario 2 memiliki rata-rata latency yang dihasilkan sebesar 8,58 ms.

$$Perbandingan = \frac{skenario\ 1}{skenario\ 2}$$
$$= 6{,}54 - 8{,}58$$
$$= 1{,}31\ ms$$

Sehingga perbandingan *latency* yang dimiliki untuk skenario 1 dan skenario 2 sebesar 1.31 ms kedua skenario untuk melakukan pengiriman data

BAB V PENUTUP

5.1 Kesimpulan

Dari proses perancangan, implementasi, serta pengujian dapat ditarik kesimpulan yaitu:

1. Rancang bangun serta implementasi yang dibuat dapat membangun sistem pemantauan lingkungan menggunakan topologi *mesh* dan mikrokontroller NodeMCU ESP8266. skenario 1 dapat dijalankan dengan baik dan menunjukkan kinerja yang dikirim dan menangkap data masing-masing mesh sesuai dengan jenis pengukuran kualitas lingkungannya masing-masing. Perangkat yang terdistribusi untuk skenario 1 mampu mem-*broadcast* pesan yang masuk pada masing-masing mesh sesuai jenis pengukuran dengan baik. Skenario 2 mampu melakukan pengiriman yang bersamaan walaupun memiliki pengukuran data untuk kualitas lingkungan berbeda-beda, data pengukuran yang tersimpan pada server sesuai dengan topik pesan yang dirancang dan telah disesuaikan sebelumnya pada masing-masing node tiap-tiap ESP-MESH
2. Hasil pengujian parameter pengukuran data skenario satu diperoleh hasil rata-rata *throughput publish* sebesar 20.75 Mbps, *throughput subscriber* sebesar 20.86 Mbps dan *latency* sebesar 6.54 ms
3. Hasil pengujian parameter pengukuran data skenario dua rata-rata *throughput publish* sebesar 20.75 Mbps, *throughput subscriber* sebesar 20.86 Mbps dan *latency* sebesar 8.58 ms

4. Pengukuran *throughput* dan *latency* menggunakan *size*/*record* untuk masing-masing *throughput* dan *latency* sebesar 8, 16, 32, 64, 128, 256, 512 dan 1024 *bytes*

Sehingga hasil dari analisa yang didapatkan dengan memerhatikan keunggulan dan kelemahan masing-masing skenario serta mengingat data yang diperoleh memiliki indeks yang sama baik dan mampu melakukan pengiriman serta pendistribusian pesan data ke server. Skenario 1 dan 2 pengaplikasian *throughput* dan *latency* pada masing-masing skenario memiliki akumulasi rata-rata *throghput* dengan indeks 4 dengan mengikuti standar **Telecommunications and Internet Protocol Harmonization Over Networks** (TIPHON). Perbandingan *throughput publish* untuk 2 skenario sebesar 1.74 Mbps dan throughput subscriber sebesar 1.76 Mbps dalam melakukan pengiriman data dan Perbandingan *latency* yang dimiliki untuk skenario 1 dan skenario 2 sebesar 1.31 ms kedua skenario untuk melakukan pengiriman data

5.2 Saran

1. Untuk pengembangan selanjutnya dapat dilakukan pengembangan performa dan akurasi yang lebih tinggi untuk size node per bytes yang ada pada tiap node node agar data dapat dijangkau lebih luas tanpa mengurangi performa tingkat akurasi data baik dalam satuan detik.
2. Diharapkan pengembangan selanjutnya dapat diterapkan pada bidang-bidang yang membutuhkan pengukuran data tertentu. seperti untuk skenario satu dapat diaplikasikan ke tempat terpusat seperti rumah sakit, gedung atau instansi tertentu dan skenario dua dapat diaplikasikan ke wilayah seperti pertanian, wilayah perkotaan & pedesaan atau tempat yang memerlukan pendeteksian untuk pemantauan lingkungan yang variatif dan luas
3. Sistem *Implementasi* jaringan mesh ini memiliki fitur-fitur yang belum sempurna. Oleh karena itu, hal ini dapat menjadi acuan untuk dikembangkan kedepannya agar lebih bermanfaat.

DAFTAR PUSTAKA

Adafruit, Learing, & System. (2018). *TSL2561 Luminosity Sensor*. 1–17.

Astrophysics and Space Science Library. (2020). In *Astrophysics and Space Science Library0067-0057* (Vol. 378, Issue Chapter 8). http://www.springerlink.com/index/10.1007/978-1-4614-0580-1_8%5Cnpapers2://publication/doi/10.1007/978-1-4614-0580-1_8

BPIW. (2019). *THE NATIONAL URBAN DEVELOPMENT PROJECT ENVIRONMENTAL AND SOCIAL MANAGEMENT FRAMEWORK*. *February*.

Fajrika hadnis Putra, R., Muslim Lhaksmana, K., & Adytia, D. (2018). Aplikasi IoT untuk Rumah Pintar dengan Fitur Prediksi Cuaca. *Ind.Journalon Computinng*, *5*(1), 1746–1760.

Green. (2018). Topologi Jaringan. *Topologi Jaringan Topologi Jaringan*, 6–22.

Haseeb, K., Din, I. U. D., & Member, S. (2020). RTS : A Robust and Trusted Scheme for IoT-Based Mobile Wireless Mesh Networks. *IEEE Access*, *8*, 68379–68390. https://doi.org/10.1109/ACCESS.2020.2985851

Hestylesta. (2009). *Bab ii teori penunjang 2.1 umum*. *September 2015*, 6–26.

Indrayani, S. A. (2018). PENCEMARAN UDARA AKIBAT KINERJA LALU-LINTAS Air Pollutions Due to Traffic Performance of Motor Vehicles in Medan City. *Jurnal Pemukiman*, *13*(1), 13–20. http://103.12.84.119/index.php/JP/article/view/274

Jaladi, A. R., Khithani, K., Pawar, P., Malvi, K., & Sahoo, G. (2017). Environmental Monitoring Using Wireless Sensor Networks (WSN) based on IOT . *International Research Journal of Engineering and Technology (IRJET)*, *4*(1), 1371–1378. https://irjet.net/archives/V4/i1/IRJET-V4I1246.pdf

Jayani, D. H. (2020). Berapa Jumlah Penduduk Perkotaan di Indonesia ? *Databoks*, 2025. https://databoks.katadata.co.id/datapublish/2019/09/11/berapa-jumlah-penduduk-perkotaan-di-indonesia

Jiang, X., Zhang, H., Barsallo Yi, E. A., Raghunathan, N., Mousoulis, C., Chaterji, S., Peroulis, D., Shakouri, A., & Bagchi, S. (2021). Hybrid Low-Power Wide-Area Mesh Network for IoT Applications. *IEEE Internet of Things Journal*, *8*(2), 901–915. https://doi.org/10.1109/JIOT.2020.3009228

Laghari, A. A., Wu, K., Laghari, R. A., Ali, M., & Khan, A. A. (2022). A Review and State of Art of Internet of Things (IoT). *Archives of Computational Methods in Engineering*, *29*(3), 1395–1413. https://doi.org/10.1007/s11831-021-09622-6

Light, R. A. (2017). Mosquitto: server and client implementation of the MQTT protocol. *The Journal of Open Source Software*, *2*(13), 265. https://doi.org/10.21105/joss.00265

Lin, H. H., Tsai, H. Y., Chan, T. C., Huang, Y. S., Chu, Y. S., Chen, Y. C., Liao, T. S., Fang, Y. M., Lee, B. J., & Lee, H. C. (2015). An open-source wireless mesh networking module for environmental monitoring. *Conference Record - IEEE Instrumentation and Measurement Technology Conference*, *2015-July*, 1002–1007. https://doi.org/10.1109/I2MTC.2015.7151407

Marlita, D., Nlql, V., Gdq, N., Dnledw, N., Whumdglq, G., Xgdud, D. S., & Nhqgduddq, N. (2019). *Pencemaran Udara Akibat Emisi Gas Buang Kendaraan Bermotor*. *01*(03).

Nathan, A. J., & Scobell, A. (2012). Implementasi Wireless Network Sensor. *Foreign Affairs*, *91*(5), 1689–1699.

Pascale, E. Di, Macaluso, I., Nag, A., Kelly, M., & Doyle, L. (2018a). *Komputasi Terdistribusi melalui Jaringan IoT Mesh*. *4662*(ICC), 1–13. https://doi.org/10.1109/JIOT.2018.2823978

Pascale, E. Di, Macaluso, I., Nag, A., Kelly, M., & Doyle, L. (2018b). *The Network as a Computer : a Framework for Distributed Computing over IoT Mesh Networks*. *4662*(c), 1–13. https://doi.org/10.1109/JIOT.2018.2823978

Rajkhowa, R. (2012). Light Pollution and Impact of Light Pollution. *International Journal of Science and Research (IJSR) ISSN (Online Impact Factor*, *3*(10), 2319–7064. www.ijsr.net

Rawat, P., Singh, K. D., Chaouchi, H., & Bonnin, J. M. (2014). Wireless sensor networks: A survey on recent developments and potential synergies. *Journal of Supercomputing*, *68*(1), 1–48. https://doi.org/10.1007/s11227-013-1021-9

Rizzardi, A., Sicari, S., Miorandi, D., & Coen-Porisini, A. (2016). AUPS: An Open Source AUthenticated Publish/Subscribe system for the Internet of Things. *Information Systems*, *62*, 29–41. https://doi.org/10.1016/j.is.2016.05.004

Sabiq, A., Nurmaya, ., Alfarisi, T., & Pratama, Y. A. (2017). Purwarupa Sistem Pemantauan Kualitas Udara dan Cuaca Melalui Web Berbasis Wireless Sensor Network. *JST (Jurnal Sains Dan Teknologi)*, *6*(2), 248. https://doi.org/10.23887/jst-undiksha.v6i2.11250

Samiaji, T. (2021). Gas CO2 di wilayah Indonesia. *Berita Dirgantara*, *12*(2), 68–75.

Saputra, Y. E. (2018). *Rancang Bangun Wireless Sensor Network untuk monitoring pencemaran air sungai menggunakakn topologi mesh network*. *1*.

Sensor, H. (n.d.). *Temperature Sensor*.

Sulaiman. (2016). *mesh*.

Surakarta, U. M. (2010). *BAB I*. 1–18.

TIPHON. (2002). Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON); General aspects of Quality of Service (QoS). *Etsi Tr 101 329 V2.1.1*, *1*, 1–37.

Tjaronge, P. D. E. H. M. W. (2016). *TERHADAP IKLIM MIKRO DI KOTA MAKASSAR*. 1–12.

v. M. buyanov. (2017). Mesh. *Angewandte Chemie International Edition, 6(11), 951–952.*, 5–44.

Wählisch, M. (2014). *22 . Modeling the Network Topology* (Issue January). https://doi.org/10.1007/978-3-642-12331-3

Winseng. (2015). *MQ135 Semiconductor Sensor for Air Quality Control*. 2–4. http://www.china-total.com/Product/meter/gas-sensor/MQ135.pdf

Espressif. (n.d.). *ESP-NOW - ESP32 - ESP-IDF Programming Guide latest documentation*.https://docs.espressif.com/projects/espidf/en/latest/esp32/apireference/network/esp_now.html

L A M P I R A N

Lampiran 1. *Source Code masing*-masing ESP-MESH tiap *node*

ESP-DHT22, ESP-MQ135, ESP-TSL2561

```
#include <Arduino_JSON.h>
#include <DHT.h>

#include "TSL2561.h"
#include "MQ135.h"
#include "painlessMesh.h"

// MESH Details
#define   MESH_PREFIX     "whateverYouWant"
#define   MESH_PASSWORD   "somethingSneaky"
#define   MESH_PORT       5555

#define DHTPIN 5
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);

#define ANALOGPIN A0
MQ135 gasSensor(ANALOGPIN);
TSL2561 tsl(TSL2561_ADDR_FLOAT);

//Number for this node
int nodeNumber = 1-9;

//String to send to other nodes with sensor readings
String readings;

Scheduler userScheduler; // to control your personal
task
painlessMesh  mesh;

// User stub
void sendMessage() ; // Prototype so PlatformIO
doesn't complain
String getReadings(); // Prototype for sending sensor
readings


String getReadings () {
  JSONVar jsonReadings;
  jsonReadings["id"] = nodeNumber;
  jsonReadings["hum"] = dht.readHumidity();
  jsonReadings["temp"] = dht.readTemperature();
jsonReadings["ppm"] = gasSensor.getPPM();

jsonReadings["lux"]=
tsl.getLuminosity(TSL2561_INFRARED);
```

```
  readings = JSON.stringify(jsonReadings);
  return readings;
  }

//Create tasks: to send messages and get readings;
Task taskSendMessage(TASK_SECOND * 1 , TASK_FOREVER,
&sendMessage);

void sendMessage () {
  String msg = getReadings();
  mesh.sendBroadcast(msg);
}

// Needed for painless library
void receivedCallback( uint32_t from, String &msg ) {
  Serial.printf("Received from %u msg=%s\n", from,
msg.c_str());
}

void newConnectionCallback(uint32_t nodeId) {
  Serial.printf("New Connection, nodeId = %u\n",
nodeId);
}

void changedConnectionCallback() {
  Serial.printf("Changed connections\n");
}

void nodeTimeAdjustedCallback(int32_t offset) {
  Serial.printf("Adjusted time %u. Offset = %d\n",
mesh.getNodeTime(),offset);
}

void setup() {
  Serial.begin(115200);
  dht.begin();
  randomSeed(42);
```

Lampiran 2. *Source Code* ESP-MESH ke *Bridge* Topologi I

MESH BRIDGE- (DHT22, MQ135, TSL2561)

```
#include <Arduino.h>
#include <painlessMesh.h>
#include <PubSubClient.h>
#include <WiFiClient.h>

#define   MESH_PREFIX     "whateverYouWant"
#define   MESH_PASSWORD   "somethingSneaky"
#define   MESH_PORT       5555

#define   STATION_SSID     "LAB_TA"
#define   STATION_PASSWORD "t4ny4h3ndr4"

#define HOSTNAME "MQTT_Bridge2"

// Prototypes
void receivedCallback( const uint32_t &from, const String
&msg );
void mqttCallback(char* topic, byte* payload, unsigned
int length);

IPAddress getlocalIP();

IPAddress myIP(0,0,0,0);
IPAddress mqttBroker(192, 168, 43, 12);

painlessMesh  mesh;
WiFiClient wifiClient;
PubSubClient mqttClient(mqttBroker, 1883, mqttCallback,
wifiClient);

void setup() {
  Serial.begin(115200);

s  mesh.setDebugMsgTypes( ERROR | STARTUP | CONNECTION );
// set before init() so that you can see startup messages

  // Channel set to 6. Make sure to use the same channel
for your mesh and for you other
  // network (STATION_SSID)
```

```
    mesh.init( MESH_PREFIX, MESH_PASSWORD, MESH_PORT,
  WIFI_AP_STA );
    mesh.onReceive(&receivedCallback);

    mesh.stationManual(STATION_SSID, STATION_PASSWORD);
    mesh.setHostname(HOSTNAME);
    mesh.setRoot(true);
    mesh.setContainsRoot(true);
  }

  void loop() {
    mesh.update();
    mqttClient.loop();

    if(myIP != getlocalIP()){
      myIP = getlocalIP();
      Serial.println("My IP is " + myIP.toString());

      if (mqttClient.connect("painlessMeshClient")) {

  mqttClient.publish("painlessMesh/from/gateway","Ready!");
       mqttClient.subscribe("painlessMesh/to/#");
      }
    }
  }

  void receivedCallback( const uint32_t &from, const String
  &msg ) {
    Serial.printf("bridge: Received from %u msg=%s\n",
  from, msg.c_str());
    String topic = "all_node3";
    mqttClient.publish(topic.c_str(), msg.c_str());
  }
void mqttCallback(char* topic, uint8_t* payload, unsigned
int length) {
    char* cleanPayload = (char*)malloc(length+1);
    memcpy(cleanPayload, payload, length);
    cleanPayload[length] = '\0';
    String msg = String(cleanPayload);
    free(cleanPayload);

    String targetStr = String(topic).substring(16);

    if(targetStr == "gateway")
    {
      if(msg == "getNodes")
      {
```

```
      auto nodes = mesh.getNodeList(true);
      String str;
      for (auto &&id : nodes)
        str += String(id) + String(" ");
      mqttClient.publish("painlessMesh/from/gateway",
str.c_str());
    }
  }
  else if(targetStr == "broadcast")
  {
    mesh.sendBroadcast(msg);
  }
  else
  {
    uint32_t target = strtoul(targetStr.c_str(), NULL,
10);
    if(mesh.isConnected(target))
    {
      mesh.sendSingle(target, msg);
    }
    else
    {
      mqttClient.publish("painlessMesh/from/gateway",
"Client not connected!");
    }
  }
}

IPAddress getlocalIP() {
return IPAddress(mesh.getStationIP());
}
```

Lampiran 3. *Source Code* Bridge MQTT Python Server Topologi 1 & 2

```
import paho.mqtt.client as mqtt
import json
import requests
from datetime import datetime


mqtt_broker_ip = "192.168.43.12"
client = mqtt.Client()

def on_connect(client, userdata, flags, rc):
    print("Connected with result code "+str(rc))
    client.subscribe("Mytopic")

client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message
client.username_pw_set("pi", "123")
client.connect(mqtt_broker_ip, 1883)
client.loop_forever()
client.disconnect()
```

Lampiran 4. Proses hasil penerimaan data antara masing-masing board mesh pada *server*

```
{"id":1,"lux":6}
{"id":2,"lux":12}
{"id":7,"hum":70.800003051757812,"temp":30.100000381469727}
{"id":6,"ppm":13683.4951171875}
{"id":5,"ppm":23669.6875}
{"id":3,"lux":12}
{"id":4,"ppm":4805.71826171875}
{"id":9,"hum":71.199996948242188,"temp":29.600000381469727}
{"id":1,"lux":6}
{"id":7,"hum":70.800003051757812,"temp":30.100000381469727}
{"id":2,"lux":13}
{"id":3,"lux":12}
{"id":6,"ppm":13683.4951171875}
{"id":5,"ppm":23669.6875}
{"id":7,"hum":70.699996948242188,"temp":30.100000381469727}
{"id":4,"ppm":4805.71826171875}
{"id":9,"hum":71.0999984741211,"temp":29.600000381469727}
{"id":1,"lux":6}
```

```
Connected.. 0
{"id":2,"lux":2}
{"id":7,"hum":75.5,"temp":31.200000762939453}
{"id":1,"lux":65535}
{"id":5,"ppm":255784.3125}
{"id":9,"hum":78.4000015258789,"temp":30.100000381469727}
{"id":8,"hum":77.800003051757812,"temp":30.5}
{"id":3,"lux":2}
{"id":6,"ppm":21356.580078125}
{"id":2,"lux":2}
{"id":7,"hum":75.5,"temp":31.200000762939453}
{"id":1,"lux":65535}
{"id":3,"lux":3}
{"id":5,"ppm":258974.890625}
{"id":9,"hum":78.4000015258789,"temp":30}
{"id":8,"hum":77.800003051757812,"temp":30.5}
{"id":6,"ppm":23810.5625}
{"id":2,"lux":2}
{"id":7,"hum":75.5,"temp":31.200000762939453}
{"id":1,"lux":65535}
{"id":3,"lux":2}
{"id":5,"ppm":258974.890625}
{"id":9,"hum":78.4000015258789,"temp":30}
{"id":8,"hum":77.800003051757812,"temp":30.5}
{"id":6,"ppm":24070.60546875}
{"id":2,"lux":2}
```