REPOSITOR, Vol. 2, No. 1, Januari 2020, Pp. 15-26
ISSN : 2714-7975
E-ISSN : 2716-1382

Rancang Bangun Control and Monitoring Sensor Node WSN Menggunakan Protokol Message Queue Telemetry Transport

Inung Bagus Prasetyo[*1], Mahar Faiqurahman[2], Zamah Sari[3]
[1,2,3]Teknik Informatika/Universitas Muhammadiyah Malang
inungbagusp@gmail.com[*1], mahar@umm.ac.id[2] zamahsari@umm.ac.id[3]

Abstrak

Jaringan WSN semakin mendapat perhatian karena potensi solusi baru dan menarik dibidang otomasi industri, pengolahan asset, pemantauan lingkungan dan lain-lain. Namun ada beberapa masalah dalam membangun jaringan WSN. Sensor node harus berukuran kecil, hemat biaya, sumber daya rendah, sensor node harus dikelola dengan baik serta jaringan WSN memerlukan protokol komunikasi tambahan. Mengelola jaringan WSN, dapat dilakukan dengan mengontrol sensor node seperti memberikan perintah pada aktifitas pengiriman data serta melakukan monitoring untuk mengetahui kondisi sensor node. Protokol MQTT dengan jenis komunikasi publish/subscribe didesain dengan karakteristik yang hampir mirip dengan karakteristik jaringan WSN yaitu sederhana, ringan, hemat energi dan mudah untuk diimplementasikan. Pada penelitian ini akan diimplementasikan sebuah server yang dapat melakukan proses control dan monitoring sensor node jaringan WSN menggunakan protokol MQTT. Kemudian digunakan parameter RTT, parameter QoS meliputi Delay, Jitter, Throughput, Packet Loss dan parameter kondisi memori sensor node saat menggunakan protokol MQTT. Skenario RTT dan QoS menggunakan variasi ukuran data 16,32,48,64,80 dan 96 byte. Monitoring kondisi memori sensor node, dilakukan selama 1 menit dengan total 30 pengiriman data. Hasil pengujian parameter RTT cukup stabil. Pengujian QoS sangat baik dengan delay, jitter yang stabil, throughput yang terus meningkat, dan 0% data hilang pada pengujian packet loss. Pengujian kondisi memori sensor node, menunjukan hasil yang tidak menentu.

Kata Kunci: *WSN, MQTT, RTT, QoS, Memori*

Abstract

The WSN network is increasingly gaining attention because of the potential for new and attractive solutions in the field of industrial automation, asset processing, environmental preparation and others. But there are some problems in building the WSN network. Sensor nodes must manage small, cost-effective, low-resource, node sensors must be managed properly with WSN networks Managing WSN networks, can be done by controlling sensor nodes such as giving commands to data transmission activities and monitoring to determine the condition of sensor nodes. The MQTT protocol with the type of publishing / subscription communication is designed with characteristics similar to the characteristics of the WSN network, which are simple, lightweight, energy efficient and easy to implement. In this study, a server will be implemented that can control and monitor the sensors of the WSN network node using the MQTT protocol. Then RTT parameters are used, Qo parameters include Delay, Jitter, Throughput, Packet Loss and sensor node memory condition parameters when using the MQTT protocol. The RTT and QoS scenarios use variations in data size of 16,32,48,64,80 and 96 bytes. Monitoring memory nodes, performing for 1 minute with a total of 30 data transmissions. The RTT parameter testing results are quite stable. QoS testing is very good with stable delay, jitter, increasing throughput, and 0% of data lost when packet loss testing. The Test Memory node sensor, shows erratic results.

Keywords: *WSN, MQTT, RTT, QoS, Memory*

1. Pendahuluan

Wireless Sensor Network (WSN) adalah jaringan yang dibentuk oleh sejumlah besar sensor node di mana setiap node dilengkapi dengan sensor untuk mendeteksi fenomena fisik seperti cahaya, panas, tekanan, dll. Sensor node ini harus memenuhi persyaratan seperti berukuran kecil, hemat biaya dan energi yang efisien. Node yang digunakan dalam jaringan WSN

harus dilengkapi dengan sensor yang tepat, sumber daya komputasi dan memori yang diperlukan untuk pemrosesan data. Dengan kondisi *low budget*, jaringan WSN seringkali menjadi pilihan utama dalam membangun sebuah jaringan yang ditujukan untuk melakukan *monitoring* terhadap suatu lingkungan tertentu [6].

Jaringan *Wireless Sensor Network* (WSN) salah satu teknologi paling menjanjikan dan menarik perhatian belakangan ini. Wireless Sensor Network (WSN) semakin mendapat perhatian, baik dari sisi komersial maupun teknis karena potensi solusi baru dan menarik dibidang otomasi industri, pengolahan asset, pemantauan lingkungan, transportasi bisnis, dan lain-lain. Banyak dari aplikasi yang dibuat dengan memanfaatkan jaringan WSN, memerlukan protokol komunikasi sebagai penunjang kebutuhan komunikasi data yang diperlukan jaringan WSN [1].

Tidak sedikit penelitian yang menggunakan protokol komunikasi didalam jaringan WSN. Pada tahun 2014 Ping Wang melakukan sebuah penelitian dengan memperluas area WSN berdasarkan IPv6 menggunakan protokol XMPP. Protokol XMPP digunakan untuk mengoptimalkan node sensor dengan mempertimbangkan sumber daya yang terbatas pada jaringan WSN. Penelitian Ping Wang menggunakan UDP sebagai mekanisme transmisi yang mendasari untuk XMPP bukan mekanisme TCP tradisional. Dengan tujuan mengurangi *overhead* pada saat pelaksanaan, Ping Wang mengembangkan perangkat lunak klien XMPP melalui metode penguraian *frame*. Meskipun hasil eksperimen menunjukan bahwa Menggunakan XMPP melalui UDP layak diterapkan, namun node sensor terbebani oleh protokol XMPP yang terpasang pada klien karena besarnya memori yang dipakai oleh protokol XMPP [9].

Pada bulan Agustus tahun 2013, Zuo Chen melakukan sebuah penelitian dengan menggabungkan jaringan WSN dan protokol *cluster routing*. Tujuan dari makalah ini adalah untuk menggambarkan protokol *cluster* LEACH-PF, yang merupakan algoritma routing multihop dengan bidang potensial energi yang diminimalkan dari kelompok *cluster* yang terbagi. Di LEACH-PF, jaringan dibagi menjadi beberapa subjaringan dan setiap subjaringan memiliki kepala *cluster*. Kelompok-kelompok ini membangun pohon *routing* antar jalur sesuai dengan perbedaan potensial dari bidang ekuipotensial yang berbeda. Hasil simulasi menunjukkan bahwa LEACH-PF dapat mengurangi konsumsi energi jaringan secara efektif dan memperpanjang masa pakai jaringan. Tetapi setiap node sensor terbebani oleh penyimpanan clustering yang terbagi menjadi beberapa kelompok jaringan di setiap kepala clusternya [10].

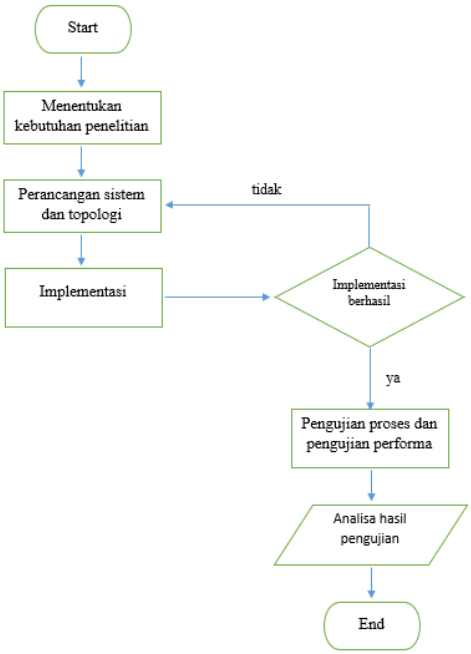
Protokol *Message Queue Telemetry Transport* (MQTT), dapat menjawab permasalahan yang terjadi pada penelitian sebelumnya. Protokol MQTT adalah sebuah protokol komunikasi yang didesain dengan karakteristik yang hampir mirip dengan karakteristik jairngan WSN, sederhana, ringan, hemat energi dan mudah untuk diimplementasikan. Dengan demikian protokol MQTT dapat diimplementasikan pada perangkat dengan sumber daya komputasi yang terbatas[3]. Protokol MQTT dapat menunjang fasilitias komunikasi jaringan WSN dengan menggunakan jenis komunikasi *publish/subscribe* tanpa membebani node sensor yang berjalan [8].

Pada penelitian sebelumnya jaringan WSN telah dibangun di kota bagian selatan Indiana dengan skala metropolitan [7]. Perancangan jaringan WSN pada skala metropolitan membuat penyebaran node memiliki skalabilitas yang luas juga. Keadaan ini menyebabkan control dan monitoring dari setiap node akan memakan waktu yang tidak efisien karena harus melakukan control dan monitoring secara langsung atau datang langsung ke setiap node WSN. Menerapkan control dan monitoring secara terpusat terhadap sistem Wireless Sensor Network penting dilakukan, karena mengingat penyebaran node yang menyebar dan memilki skalabilitas yang luas. Sedangkan dari beberapa penelitian yang telah dilakukan control dan monitoring menjadi sebuah fitur aplikasi yang digunakan untuk memonitor suatu lingkungan tertentu, tidak untuk mengontrol dan memonitor sistem WSN itu sendiri [2] [4] [7] .

Berdasarkan latar belakang diatas akan diimplementasikan control dan monitoring pada sensor node jaringan WSN (Wireless Sensor Network) menggunakan protokol MQTT. Protokol MQTT akan dapat menunjang kebutuhan fasilitas komunikasi yang dibutuhkan oleh jaringan WSN. Protokol MQTT digunakan sebagai media pengiriman data untuk melakukan proses control dan monitoring sensor node pada jaringan WSN.

2. Metode Penelitian

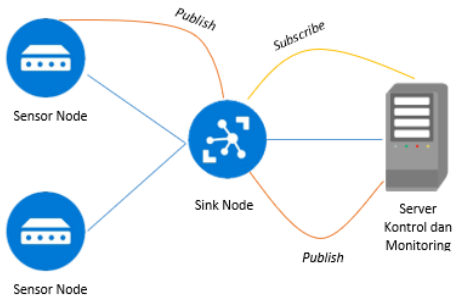
Pada penelitian ini digunakan metode penelitian yang disajikan dalam bentuk *flowchart* seperti pada Gambar 1.



Gambar 1. Flowchat Metode Penelitian

2.1 Perancangan Topologi Jaringan

Pada penelitian ini digunakan module ESP8266 12-E sebagai module komunikasi dan mikrokontroler, sensor DHT 11 sebagai media uji untuk mendapatkan data real-time suhu. Server dapat menggunakan laptop maupun PC untuk melakukan proses control dan monitoring sesuai dengan kebutuhan. Untuk dapat berkomunikasi saat proses *request* maupun *response* dengan perangkat sensor node atau perangkat lainnya, protokol MQTT yang berada di server harus terkoneksi dengan broker MQTT yang sudah terpasang pada sink node. Kemudian komunikasi data secara *real-time* akan dijembatani oleh protokol MQTT menggunakan jenis komunikasi *publish /subscribe*. Komunikasi antara sensor node dan sink node menggunakan koneksi WiFi, selain itu pada penelitian ini komunikasi antara sink node dan server *control* dan *monitoring* diasumsikan menggunakan koneksi WiFi pada jaringan lokal. Gambar 2 berikut merupakan rancangan topologi yang akan digunakan.



Gambar 2. Perancangan Topologi Jaringan WSN

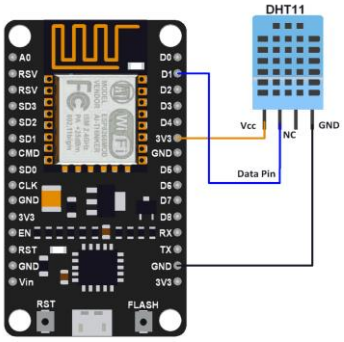
2.2 Perangkat Keras Jaringan

Terdapat beberapa perangkat keras yang digunakan dalam penelitian jaringan WSN ini, kebutuhan perangkat keras seperti pada Tabel 1.

Tabel 1. Kebutuhan Perangkat Keras

Laptop Core i3	Sebagai server control dan monitoring
NodeMCU ESP8266 12-E	Sebagai sensor node
Raspberry Pi 3 Model B	Sebagai sink node
Sensor DHT 11	Sebagai media uji sensing suhu

2.3 Desain Perancangan Sensor Node

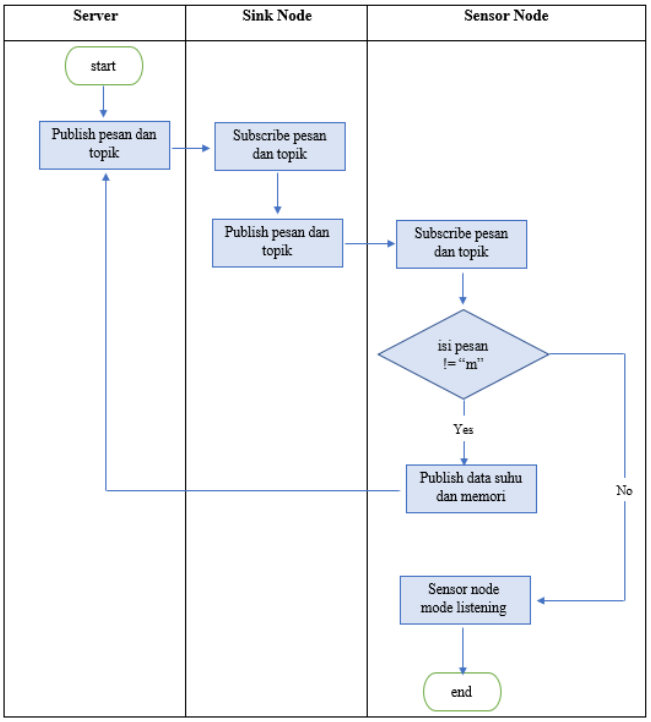


Gambar 3. Rangkaian Mikrokontroler ESP8266 12-E

Pada Gambar 3 dijelaskan rangkaian dari sensor DHT11 yang tersambung langsung dengan NodeMCU ESP8266 12-E. Sebagai media uji untuk mendapatkan data *sensing* dapat digunakan sensor apapun, namun pada penelitian ini di asumsikan menggunakan sensor DHT 11. Sensor node yang telah menerima *request* dari server akan mengirimkan data *real-time* hasil dari *sensing* suhu yang telah didapatkan dari sensor DHT11. Dengan adanya ESP8266 yang dapat tersambung langsung dengan sensor DHT11, memberikan kemudahan untuk mengakses dan memberikan intruksi melalui koneksi WiFi sebagai media perantaranya. Untuk pemrograman sensor node dapat digunakan *software* Arduino IDE dan menggunakan library yang telah tersedia agar bisa menggunakan fitur dan berkomunikasi pada perangkat dan mengendalikan sensor untuk melakukan proses *sensing*.

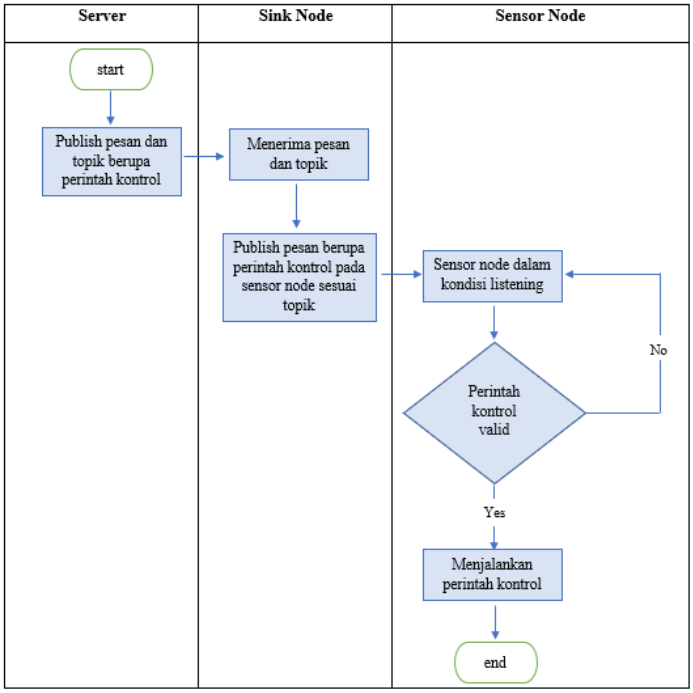
2.4 Proses Pengiriman Data

Pada Gambar 4 dijelaskan bagaimana proses pengiriman data dan perintah dari server berjalan. Pertama server mengirimkan pesan pada sink node dengan topik yang telah ditentukan. Kemudian sink node akan menerima dan meneruskan pesan pada setiap sensor node yang telah *subscribe* dengan topik tertentu. Ketika sensor node telah menerima pesan dari server, sensor node akan mengirimkan pesan kembali pada server berupa data *real-time* kondisi memori sensor node, atau sensor node akan dalam posisi *listening* menunggu pesan selanjutnya dari server.



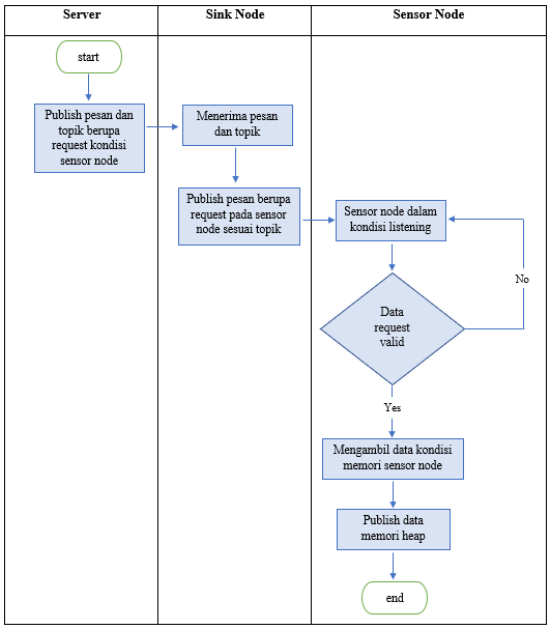
Gambar 4. Proses Pengiriman Data

2.4 Proses Kontrol dan Monitoring



Gambar 5. Proses Kontrol

Pada Gambar 5 dijelaskan tentang alur proses kontrol pada sensor node. Kondisi pertama, semua perangkat sudah saling subscribe dengan topik yang ada pada protokol MQTT. Kemudian server melakukan proses publish pesan dan topik berupa perintah untuk melakukan kontrol pada sensor node. Broker MQTT yang terpasang pada sink node akan menerima dan meneruskan pesan tersebut, pada sensor node yang telah subscribe dengan topik yang dikirimkan didalam pesan. Kemudian sensor node yang telah melakukan subscribe pada topik tertentu, akan menanggapi pesan tersebut. Jika isi pesan dinyatakan valid, maka sensor node akan menjalankan proses kontrol sesuai dengan isi pesan yang dikirimkan. Jika isi pesan dinyatakan tidak valid, maka sensor node akan kembali pada kondisi listening.



Gambar 6. Proses Monitoring

Pada Gambar 6 dijelaskan tentang alur server yang melakukan proses monitoring kondisi sistem dari sensor node. Pada proses monitoring, kondisi awal semua perangkat tetap sama dengan saat proses kontrol dilakukan. Server akan melakukan proses publish pesan dan topik berupa request pada sensor node. Broker MQTT yang terpasang pada sink node, akan menerima dan meneruskan request pada sensor node yang telah melakukan subscribe dengan topik yang dikirimkan didalam pesan. Kemudian sensor node yang telah melakukan subscribe pada topik tertentu, akan menanggapi pesan tersebut. Jika data request dinyatakan valid, maka sensor node akan mulai mengambil data kondisi memori heap dan melakukan publish data kondisi memori heap pada server. Jika data request dinyatakan tidak valid, maka sensor node akan kembali pada kondisi listening.

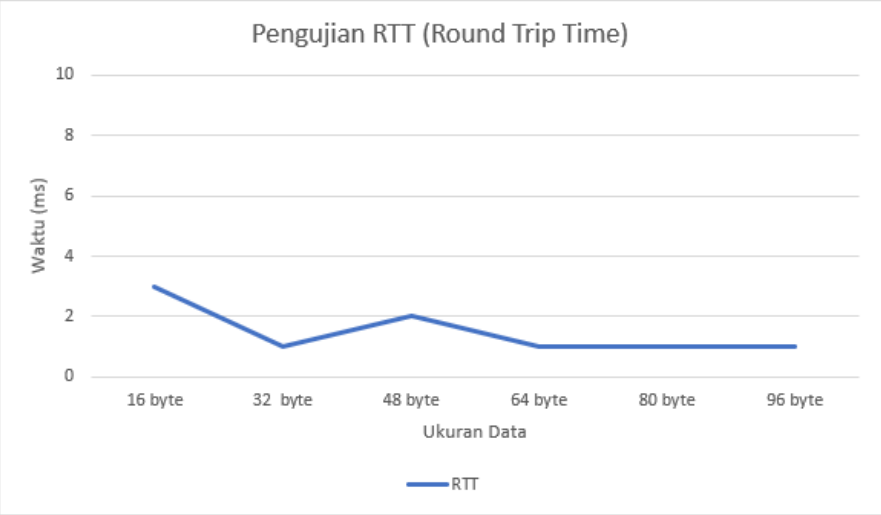
3. Hasil Penelitian dan Pembahasan

3.1. Pengujian RTT (Round Trip Time)

Pengujian RTT *(Round Trip Time)* bertujuan untuk mengetahui waktu yang dibutuhkan oleh server dan sensor node pada saat melakukan proses *request-response* menggunakan protokol MQTT. Pengujian RTT dilakukan dengan kondisi awal sensor node dan server sudah saling *subscribe* pada topik "RTT". Proses pertama server akan melakukan *request* pada sensor node dalam bentuk *publish* pesan menggunakan protokol MQTT. Setelah sensor node menerima *request* dari server, sensor node akan menanggapi *request* dengan mengirimkan *response* dalam bentuk *publish* pesan kembali pada server menggunakan protokol MQTT. Pengujian RTT dilakukan dengan variasi ukuran data yang berbeda yaitu 16,32,48,64,80 dan 96 byte.

Tabel 2. Hasil Pengujian RTT

RTT (Round Trip Time)			
Ukuran Data	Waktu Dikirim	Waktu Diterima	RTT (millisecond)
16 byte	19:49:25:277	19:49:25:280	3 ms
32 byte	19:55:34:756	19:55:34:757	1 ms
48 byte	20:00:45:246	20:00:45:248	2 ms
64 byte	20:03:29:528	20:03:29:529	1 ms
80 byte	20:06:19:335	20:06:19:336	1 ms
96 byte	20:09:29:346	20:09:29:347	1 ms



Gambar 7. Grafik Hasil Pengujian RTT

Dari hasil pengujian RTT (Round Trip Time) Tabel 2 dan Gambar 7, dapat dilihat bahwa waktu yang dibutuhkan server dan sensor node pada saat melakukan proses request-response menggunakan protokol MQTT cukup stabil. Karena, meskipun terjadi perbedaan waktu yang dibutuhkan pada saat proses request-resnponse, perbedaan waktu yang terjadi tidak begitu besar. Hasil menunjukan ada 4 variasi ukuran data yang hanya membutuhkan waktu 1 ms,

sedangkan 2 lainnya membutuhkan waktu 3ms dan 2ms. Hasil pengujian juga membuktikan bahwa ukuran data mempengaruhi waktu yang dibutuhkan pada saat proses request-response.

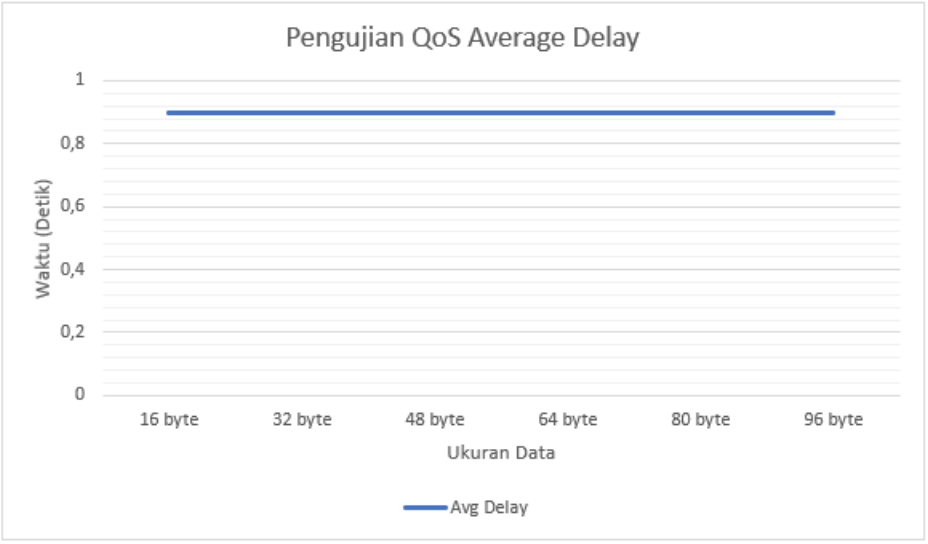
3.2. Pengujian QoS (Quality of Service)

Pengujian QoS *(Quality of Service)* bertujuan untuk mengetahui kemampuan protokol MQTT dalam menyediakan layanan komunikasi data yang baik, dengan mengatasi *jitter, delay, throughput* dan *packet loss.* Pengujian QoS (Quality of Service) dilakukan menggunakan beberapa parameter pengujian yaitu perhitungan delay, jitter, throughput dan packet loss. Pada parameter perhitungan pergeseran delay pengiriman data, sensor node akan melakukan proses publish data pada server dengan jeda waktu 1 detik sebanyak 10 kali. Pengujian dilakukan menggunakan variasi ukuran data yang berbeda yaitu, 16 byte, 32 byte, 48 byte, 64 byte, 80 byte dan 96 byte.

Setelah mendapatkan hasil pergeseran jeda waktu dari pengujian delay, tahap selanjutnya adalah melakukan pengujian jitter. Perhitungan jitter menggunakan rumus (total variasi delay / total paket data - 1). Parameter selanjutnya adalah menghitung throuhput yaitu kecepatan sensor node saat mengirimkan data pada server menggunakan protokol MQTT. Pada pengujian throughput, sensor node akan mengirimkan paket data pada server selama 10 detik dengan variasi ukuran data yang berbeda yaitu, 16 byte, 32 byte, 48 byte, 64 byte, 80 byte dan 96 byte. Setelah mendapat hasil dari pengujian throughput, akan dilakukan analisis untuk mengetahui seberapa besar (bit/second) data yang mampu dikirim sensor node saat menggunakan protokol MQTT. Parameter pengujian berikutnya adalah packet loss dilakukan dengan cara, server melakukan 20 kali proses publish data pada klien sensor node menggunakan protokol MQTT dengan variasi ukuran data yang berbeda pula.

Tabel 3. Hasil Pengujian QoS Average Delay

QoS Average Delay	
Ukuran Data	Avg Delay (detik)
16 byte	0,9
32 byte	0,9
48 byte	0,9
64 byte	0,9
80 byte	0,9
96 byte	0,9



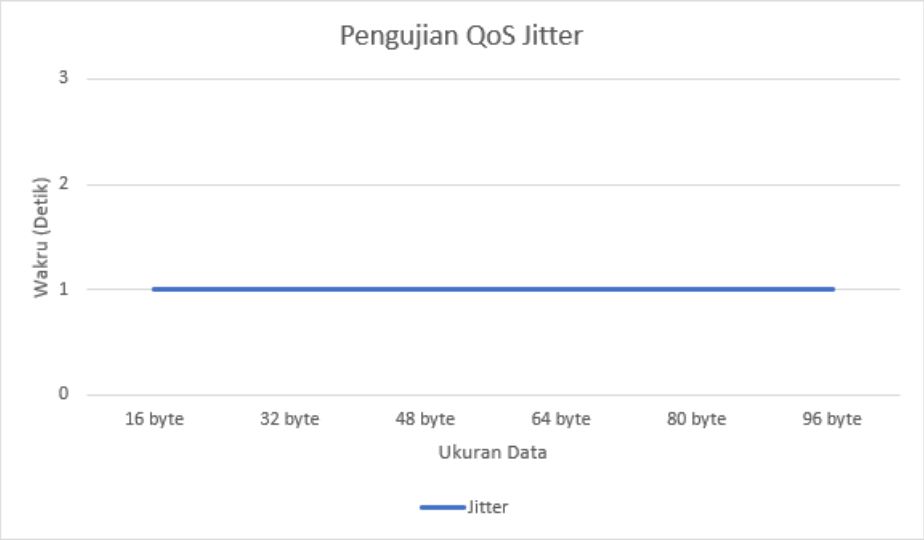
Gambar 8. Grafik Hasil Pengujian QoS Average Delay

Dari hasil pengujian delay Tabel 3 dan Gambar 8, dapat dilihat bahwa delay yang terjadi pada saat sensor node melakukan proses publish data pada server menggunakan protokol MQTT hasilnya sangat stabil. Hasil pengujian membuktikan bahwa protokol MQTT dapat menjaga delay

pengiriman data yang telah diberikan pada sensor node dengan sangat baik, ditunjukan dengan tidak adanya pergeseran delay yang terjadi pada saat pengiriman data dilakukan menggunakan protokol MQTT.

Tabel 4. Hasil Pengujian QoS Jitter

QoS Jitter	
Ukuran Data	Jitter (detik)
16 byte	1
32 byte	1
48 byte	1
64 byte	1
80 byte	1
96 byte	1



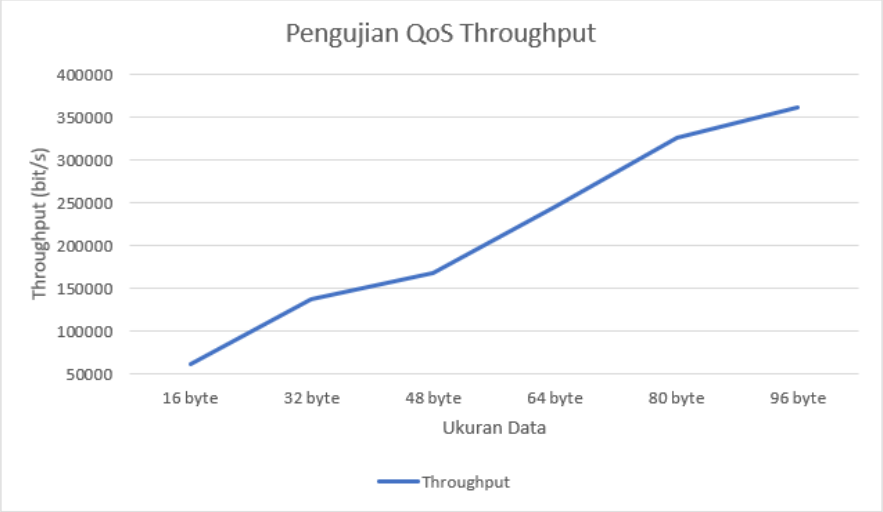
Gambar 9. Grafik Hasil Pengujian QoS Jitter

Pengujian jitter Tabel 4 dan Gambar 9 dilakukan dengan menghitung hasil pengujian delay sesuai dengan rumus. Dari hasil pengujian jitter dapat dilihat bahwa, protokol MQTT juga mampu mengatasi jitter dengan sangat baik. Terbukti dari hasil pengujian yang didapatkan, jitter sangat stabil pada waktu 1 detik.

Dengan total waktu pengiriman data yang sama, yaitu 10 detik di setiap pengiriman variasi ukuran data. Jumlah data yang dapat dikirimkan oleh sensor node pada server menggunakan protokol MQTT sangat stabil. Hal ini dibuktikan dari Tabel 5 dan Gambar 10 hasil pengujian *throughput* yang menunjukan, semakin meningkatnya jumlah data yang dapat dikirimkan oleh sensor node pada server menggunakan protokol MQTT.

Tabel 5. Hasil Pengujian QoS Throughput

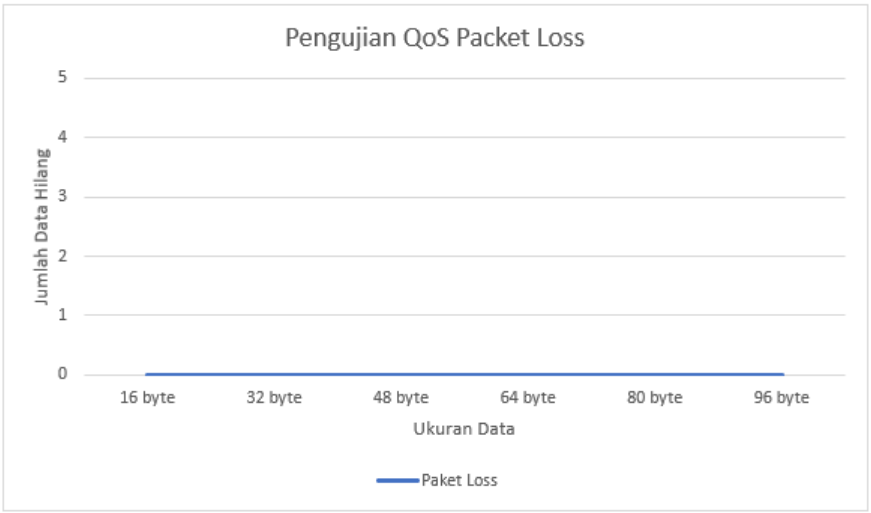
QoS Throughput			
Ukuran Data	Transfer Size (bit)	Transfer Time (second)	Throughput bit/second
16 bit	627.000	10	62.700
32 bit	1.382.400	10	138.240
48 bit	1.689.600	10	168.960
64 bit	2.457.600	10	245.760
80 bit	3.264.000	10	326.400
96 bit	3.609.600	10	360.960



Gambar 10. Grafik Hasil Pengujian QoS Throughput

Tabel 6. Hasil Pengujian QoS Packet Loss

Packet Loss				
Ukuran Data	Dikirim	Diterima	Hilang	Presentase Loss
16 byte	20	20	0	0%
32 byte	20	20	0	0%
48 byte	20	20	0	0%
64 byte	20	20	0	0%
80 byte	20	20	0	0%
96 byte	20	20	0	0%



Gambar 11. Grafik Hasil Pengujian QoS Packet Loss

Dari hasil pengujian Tabel 6 dan Gambar 11, packet loss yang dilakukan menggunakan protokol MQTT, dapat diketahui bahwa protokol MQTT dapat mengatasi pengiriman paket data dengan sangat baik. Terbukti dari hasil pengujian yang dilakukan dengan variasi ukuran data yang berbeda, tidak ada satupun paket data yang hilang pada saat proses pengiriman data dilakukan.

3.3. Pengujian Kondisi Memori Tersisa Sensor Node

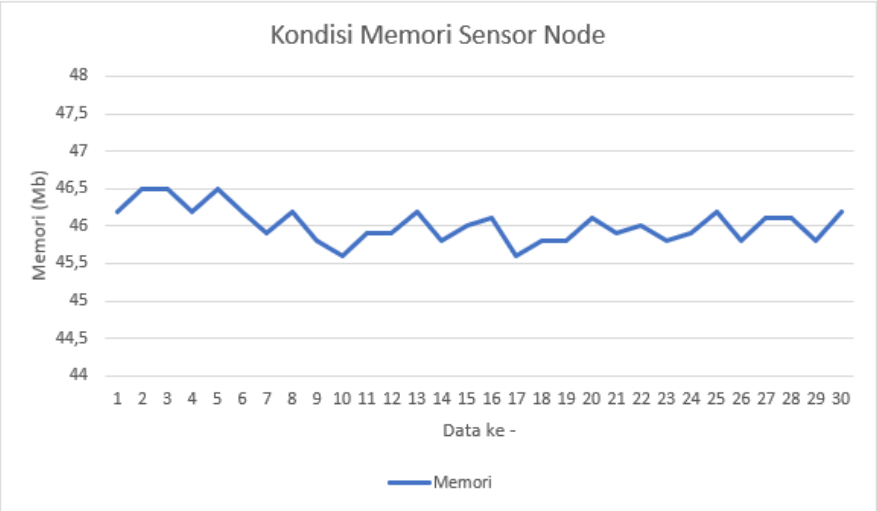
Pada pengujian ini bertujuan untuk mengetahui kondisi memori tersisa sensor node ketika menggunakan protokol MQTT untuk komunikasi data. Mengetahui kondisi memori tersisa dari

sensor node, penting untuk dilakukan agar ketika menambahkan fungsi-fungsi baru pada sensor node dapat berjalan dan tidak kehabisan memori sehingga menyebabkan proses yang dijalankan oleh sensor node menjadi lambat.

Pengujian kondisi memori tersisa sensor node dilakukan dengan cara, server mengirimkan pesan berupa perintah menggunakan protokol MQTT pada sensor node. Kemudian setelah sensor node menerima pesan dari server, sensor node akan mulai mengambil data kondisi memori tersisa dan mengirimkannya pada server. Pengiriman data kondisi memori tersisa sensor node, dilakukan selama 1 menit dengan jeda waktu pengiriman 2 detik yang berarti data kondisi memori tersisa dikirimkan pada server sebanyak 30 kali.

Tabel 7. Hasil Pengujian Kondisi Memori Tersisa Sensor Node

Kondisi Memori Tersisa Sensor Node					
Urutan Data	Memori (Mb)	Urutan Data	Memori (Mb)	Urutan Data	Memori (Mb)
Data 1	46,2	Data 11	45,9	Data 21	45,9
Data 2	46,5	Data 12	45,9	Data 22	46,0
Data 3	46,5	Data 13	46,2	Data 23	45,8
Data 4	46,2	Data 14	45,8	Data 24	45,9
Data 5	46,5	Data 15	46,0	Data 25	46,2
Data 6	46,2	Data 16	46,1	Data 26	45,8
Data 7	45,9	Data 17	45,6	Data 27	46,1
Data 8	46,2	Data 18	45,8	Data 28	46,1
Data 9	45,8	Data 19	45,8	Data 29	45,8
Data 10	45,6	Data 20	46,1	Data 30	46,2



Gambar 12. Grafik Hasil Pengujian Kondisi Memori Tersisa Sensor Node

Dari hasil pengujian Tabel 7 dan Gambar 12 yang dilakukan, dapat dilihat kondisi memori tersisa sensor node pada saat menggunakan protokol MQTT tidak stabil. Namun jika diteliti lagi, naik dan turunnya kondisi memori sensor node pada saat menggunakan protokol MQTT tidak menyebabkan proses pengiriman data menjadi lambat dan tidak menyebabkan hilangnya data yang dikirim. Karena sensor node tetap dapat mengirimkan 30 data kondisi memori tersisa pada server selama satu menit dengan jeda waktu 2 detik pada saat pengiriman data dilakukan.

4. Kesimpulan

Berdasarkan hasil pengujian dan analisis data dengan judul “Rancang Bangun Control and Monitoring Sensor Node WSN Menggunakan Protokol Message Queue Telemetry Transport (MQTT)” yang telah dilakukan, maka dapat diambil kesimpulan sebagai berikut :

1. Semua perangkat yang digunakan untuk membangun jaringan WSN seperti laptop yang terinstal ubuntu 16.04 Lts sebagai server, Raspberry Pi 3 Model B sebagai sink node dan

NodeMCU ESP8266 12-E sebagai sensor node, dapat saling berkomunkasi dengan baik dengan menggunakan protokol MQTT.

2. Analisis dari pengujian dengan menggunakan parameter RTT, menunjukan hasil yang cukup stabil. Dengan 4 waktu proses request-response yang stabil pada waktu 1ms sedangkan 2 lainnya pada waktu 2ms dan 3ms.
3. Dari analisis hasil pengujian QoS dengan parameter delay, jitter, throughput dan packet loss. Protokol MQTT dapat memberikan layanan jaringan yang baik, ditunjukan dengan hasil parameter delay dan jitter yang sangat stabil. Hasil parameter throughput ditunjukan dengan grafik yang terus meningkat menandakan bahwa, semakin banyak data yang dapat dikirimkan sensor node menggunakan protokol MQTT. Protokol MQTT juga mampu mengatasi pengiriman data dengan baik yang ditunjukan pada hasil pengujian parameter packet loss.
4. Analisis kondisi memori tersisa sensor node pada saat menggunakan protokol MQTT, menunjukan hasil yang tidak menentu. Namun dari skenario pengujian yang dibuat, hasil yang didapatkan tidak menyebabkan proses pengiriman data menjadi lambat dan tidak ada paket data yang hilang pada saat pengiriman berlangsung.

Dari hasil kesimpulan diatas, terdapat beberapa saran agar penelitian ini dapat dikembangkan lebih lanjut dan lebih baik lagi. Menambahkan variasi jarak pada saat melakukan pengujian. Menambah jumlah sensor node yang digunakan, menambahkan variasi jarak pada saat melakukan pengujian, dapat menambahkan variasi topologi dengan penambahan sink node, implementasi jaringan WSN menggunakan internet, agar proses kontrol dan monitoring lebih luas.

Referensi

[1] H. W. and Y. H. Haiqing Yang, “Architecture of Wireless Sensor Network for Monitoring Aquatic Environment of Marine Shellfish,” Asian Control Conf., vol. 7, pp. 27–29, 2009.
[2] S. A. K. Naili Shofa, Andrian Rakhmatsyah, “Infusion Monitoring using WiFi (802.11) through MQTT Protocol,” IEEE, 2017.
[3] H. J. Shinho Lee, Hyeonwoo Kim, Dong-Kweon Hong, “Correlation Analysis of MQTT Loss and Delay According to QoS Level,” IEEE, pp. 714–717, 2013.
[4] G. K. Herry Z. Kotta, Kalvein Rantelobo, Silvester Tena, “Wireless Sensor Network for Landslide Monitoring in Nusa Tenggara Timur,” TELKOMNIKA, vol. 9, no. 1, pp. 9–18, 2011.
[5] D. R. K. R.Priyanka, S.K. Thai Bhuvana, Archanaa Raveendran, “Vehicle Pollutants Control Using Sensors And Arduino,” IEEE, vol. 3, pp. 480–484, 2017.
[6] A. W. Hollger Karl, “Protocols And Architectures For Wireless Sensor Networks,” pp. 1–432.
[7] M. D. L. Luis Montestruque, “CSOnet: A Metropolitan Scale Wireless Sensor-Actuator Network,” IEEE, pp. 1–7.
[8] N. S. Sevil Ahmed, Andon Topalov, “A Robotized Wireless Sensor Network Based on MQTT Cloud Computing.”
[9] W. W. Ping Wang, Heng Wang, “Design and Implementation of XMPP for Wireless Sensor Networks Based on IPv6,” Atl. Press, pp. 852–855, 2014.
[10] R. ZuoChen, YaoXiao, XiaodongLi, “A Clustering Protocol for Wireless Sensor Networks Based on Energy Potential Field,” Sci. World J., no. 829861, pp. 1–7, 2013.

ISSN: 2714-7975; E-ISSN: 2716-1382