Implementasi TLS Sebagai Metode Keamanan Protokol Jaringan Pada MQTT Berbasis Raspberry PI

Muhammad Reihan Iswan Rafi Fauzan[1], Usman B Hanafi[2], Taufik Irfan[3]

[1,2,3]*Jurusan Teknik Elektro, Politeknik Negeri Bandung, Bandung 40012*
E-mail : muhammad.reihan.tkom19@polban.ac.id
E-mail :usmanb@polban.ac.id
E-mail : taufik.irfan@polban.ac.id

ABSTRAK

Didalam kemajuan zaman yang kini telah memasuki *society 5.0*, banyak perkembangan yang terjadi dengan sangat pesat baik dari segi inovasi atau pengembangan teknologi terdahulu. Hal ini dibuktikan oleh memungkinkannya manusia untuk bisa hidup berdampingan sehari-hari bersama ilmu pengetahuan khususnya didalam bidang IoT(*Internet of things*). Salah satu bentuk pemanfaatan IoT didalam telekomunikasi ialah terciptanya sebuah jaringan dengan nama protokol MQTT(*Message Queuing Telementry Transport*) yang diterapkan untuk konsep *smart environment*. Protokol ini merupakan jaringan protokol yang dibentuk berdasarkan model *publisher→broker→subscriber*. Dimana *publisher* bekerja sebagai pengirim pesan sementara *subscriber* sebagai penerima pesan. Adapun broker merupakan pihak ketiga didalam protokol MQTT yang tugasnya sebagai penampung dan menyalurkan pesan dari *publisher* sebelum nantinya disalurkan kepada *subscriber* ataupun sebaliknya. Kemudian berdasarkan hasil yang diperoleh dalam usaha meningkatkan keamanan dalam pengiriman pesan dilakukan pengimplementasian TLS (*Transport Layer Security*) sebagai metode protokol keamanan tambahan untuk mengamankan pesan ataupun komunikasi data didalam protokol MQTT.

Kata Kunci
Publisher, Broker, Subscriber, Internet Of Things

1. PENDAHULUAN

Dalam era peradaban digitalisasi masa kini, perkembangan teknologi dengan memanfaatkan integrasi penerapan IoT(*Internet of things*) telah banyak dilakukan disetiap bidang untuk mempermudah atau bahkan membantu keberlangsungan hidup bermasyarakat. Salah satu bentuk upaya pengembangan dan integrasi didalam bidang telekomunikasi ialah dengan menciptakan sebuah protokol jaringan komunikasi yang bernama MQTT(*Message Queuing Telementry Transport*). MQTT sendiri merupakan protokol konektivitas antara *machine to machine* atau IoT dengan berbasis *open-source* yang dirancang untuk jaringan bandwith rendah dengan latensi tinggi atau biasa digunakan sebagai solusi komunikasi untuk jaringan yang kurang baik[1].

Sebelumnya didalam bidang ilmu telekomunikasi, protokol MQTT sendiri pada mulanya ialah protokol jaringan yang diciptakan dan dikembangkan dengan tujuan dapat terintegrasi dengan IoT akan tetapi, protokol jaringan MQTT memiliki kelemahan yang cukup mencolok dikarenakan jika didalam penggunaannya digunakan *broker online* gratis dengan membuat sebuah topik atau bisa disebut dengan *password* yang cenderung bersifat umum maka, besar kemungkinan pesan ataupun paket data yang dikirimkan rentan terkena *hacking* ataupun jenis *cybercrime* lainnya dikarenakan pada broker MQTT gratis tidak memiliki pengamanan ganda atau biasa disebut dengan enkripsi data.

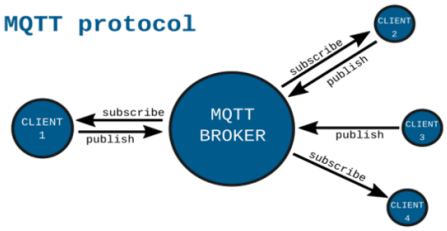
Sebenarnya, jika ditinjau dari permasalah yang terjadi terdapat solusi ataupun konklusi yang bisa menjawab kelemahan dari protokol MQTT tersebut salah satunya ialah menggunakan broker *online* berbayar untuk mengantisipasi terjadinya kebocoran data. Akan tetapi tujuan dari penelitian didalam artikel ini ialah mencoba untuk melakukan implementasi sebuah protokol keamanan atau

kriptografi dengan memanfaatkan TLS(*Transport Layer Security*) sebagai metode keamanan tambahan dalam mengenkripsi dan mengamankan data pada MQTT menggunakan *broker online* gratis.

2. PUSTAKA TERKAIT

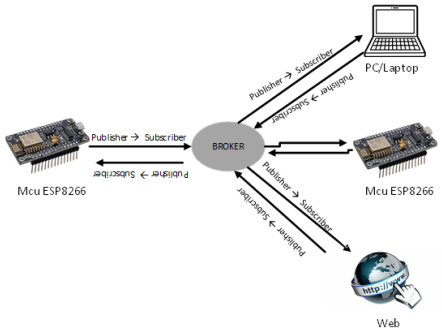
2.1 MQTT

MQTT (*Message Queuing Telementry Transport*). merupakan protokol komunikasi dalam jaringan yang berperan untuk mengirim dan menerima berupa pesan ataupun biasanya paket data yang dikombinasikan dengan IoT baik itu pada modul ESP8266 ataupun yang lainnya. MQTT sendiri juga bekerja pada TCP/IP (*Transsmision Control Protocol/Internet Protocol*). Hal ini merupakan sebuah standar yang digunakan untuk mengirim atau menerima data pada jaringan internet. [11].



Gambar 1 MQTT

Didalam protokol MQTT setiap *client* dapat menjadi *publisher* sekaligus menjadi *subscriber*. Artinya, dalam konsep protokol MQTT *client* bisa menjadi pengirim dan penerima data dalam waktu yang bersamaan seperti pada contoh gambar dibawah.



Gambar 2 Ilustrasi MQTT

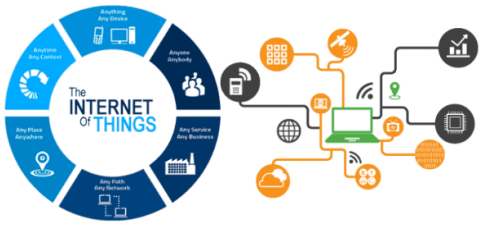
2.2 TLS (*Transport Layer Security*)

Transport Layer Security atau dikenal sebagai TLS merupakan salah satu jenis protokol jaringan yang berfungsi sebagai metode dalam pengamanan data atau enkripsi pesan transmisi data di internet. Contoh penerapan TLS ialah pengimplementasian pada website yaitu HTTPS, adapun kegunaan lain TLS juga digunakan untuk mengirim email, koneksi FTP(*File Transfer Protocol*) dan VPN(*Virtual Private Network*), serta pesan instan dan *voice over* IP. TLS biasanya digunakan terutama di daerah-daerah dimana data sensitif diperhatikan seperti pada sektor perbankan, penyimpanan data pelanggan, kata sandi, dan komunikasi digital. Tujuannya adalah memastikan transmisi data yang aman dan untuk memastikan tingkat integritas serta kredibilitas dalam penggunaan komunikasi yang aman, nyaman, dan terpercaya keamanannya. [12]

Gambar 3 TLS (*Transport Layer Security*)

2.3 *Internet Of Things* (IoT)

Internet of thigs atau disingkat IoT merupakan sebuah konsep yang diterapkan sejak masuknya era teknologi pada masa 4.0 dimana, sebuah objek memiliki kemampuan untuk mentransmisikan atau mengirimkan data melalui jaringan tanpa menggunakan bantuan dan peranan manusia.



Gambar 4 IoT

Didalam unsur mengenai IoT, setidaknya terdapat beberapa unsur dan peranan penting diantaranya ialah : [13]

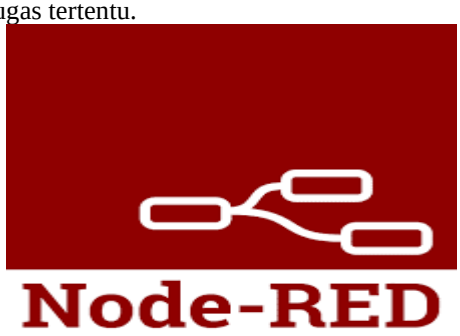
- *Artificial Intelligence* (AI)
- Konektivitas
- Sensor

2.4 Node-RED

Node-Red merupakan sebuah browser open source berbasis *flow based* atau diagram alir. Dimana, pada Node-Red biasanya digunakan sebagai tampilan berbasis website yang

didalamnya terdapat runtunan alur dari pembuatan MQTT untuk setiap *flow base* IoT yang akan digunakan. Node-RED sendiri pada MQTT digunakan sebagai media untuk pembuatan tampilan UI (*User Interface*) dari protokol komunikasi yang terintegrasi dengan IoT.

Adapun lingkungan pemrograman visualnya, Node-RED mempermudah penggunanya untuk membuat aplikasi sebagai *"flow"*. Flow ini terbentuk dari node-node yang saling berhubungan di mana tiap node melakukan tugas tertentu.



Gambar 5 Node-RED

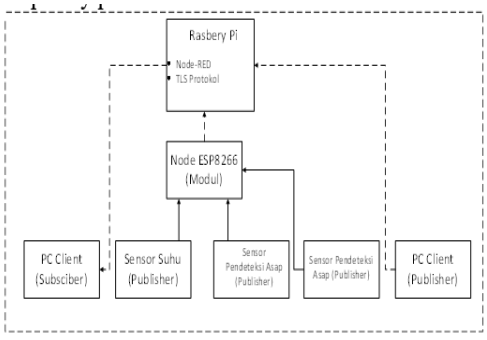
3. METODOLOGI PELAKSANAAN

3.1 Perancangan

Terdapat beberapa tahap perancangan diantaranya seperti membuat membuat blok diagram keseluruhan, perancangan skematik, diagram ilustrasi sistem, membuat *flowchart* mengenai sensor suhu, sensor pendeteksi api, dan sensor pendeteksi asap.

3.2 3.1.1 Diagram Blok

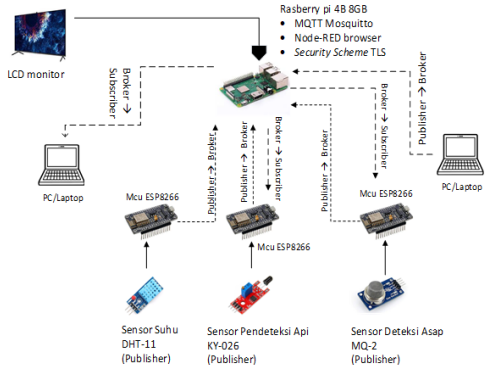
Pada diagram blok secara keseluruhan, memaparkan cara kerja dari perancangan implementasi TLS sebagai *security scheme* pada MQTT berbasis raspberry pi



Gambar 6 Diagram Blok Keseluruhan

3.1.2 Diagram Blok Ilustrasi

Dilakukan perancangan diagram ilustrasi atau cara kerja keseluruhan pada implementasi TLS sebagai *security scheme* pada MQTT berbasis raspberry pi. Untuk cara kerja, gambaran dan pemaparan blok diagram ilustrasi tersebut kurang lebih sama seperti diagram ilustrasi blok keseluruhan. Hanya saja digambarkan secara ilustrasi.



Gambar 7 Diagram Blok Ilustrasi

Pada gambar diatas terlihat blok diagram keseluruhan dari perancangan diagram blok ilustrasi. Terlihat pada gambar diatas bahwasannya digunakan tiga buah MCUESP8266 yang berperan sebagai modul untuk pengiputan program yang akan digunakan oleh sensor. Kemudian terdapat tiga buah sensor seperti sensor suhu(DHT-11), sensor pendeteksi api(KY-026), dan sensor pendeteksi asap(MQ-2). Ketiga buah sensor ini akan berperan sebagai *publisher*. Karena pada dasarnya ketiga sensor inilah yang mengirimkan data kepada raspbery pi melalui modul MCUESP8266, kemudian adapun raspbery pi disini akan berperan sebagai *broker*. Bisa dibilang, bagian paling penting yang menjadikan protokol jaringan ini disebut MQTT berada pada bagian *broker* ini. Didalam raspbery pi akan menerima pesan berupa paket data yang sebelumnya dikirimkan oleh *publisher*.

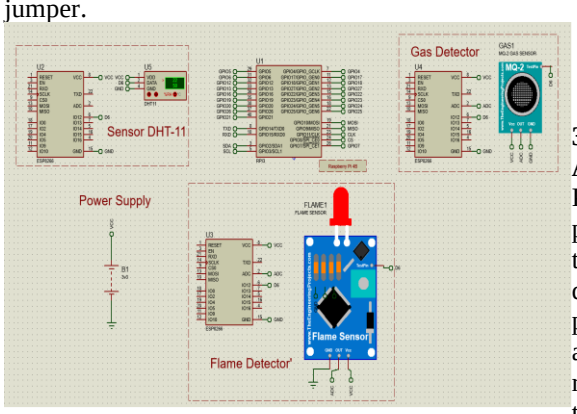
Raspberry pi sendiri selain menjadi *broker* dalam protokol MQTT ini bisa juga merangkap menjadi *subscriber*. Hal ini terjadi karena nantinya raspberry pi akan mengimplementasikan protokol MQTT mosquito dimana dalam implementasinya, protokol MQTT mosquito broker ini mampu menghubungkan atau menjadikan sebuah *device* (Raspberry pi) menjadi *publisher* atau *subscriber* dalam waktu bersamaan. Dimana, Raspberry pi dikatakan sebagai *subscriber* karena menerima hasil output ketiga buah sensor(*publisher*) berupa paket data yang dikirimkan melalui mosquito sebagai brokernya.

Untuk bagian PC sebagai *publisher* juga pada dasarnya sama, dalam pelaksanaan saat diujikan untuk mengirim pesan kepada rasbery pi (*broker*) lalu dari *broker* akan disampaikan kepada PC *client* sebagai *subscriber*. Untuk bagian TLS sebagai metode

keamanan jaringan tambahan akan diimplementasikan untuk mengenkripsi pesan diantara *publisher* dan *subscriber* pada jaringan lolcalhost raspberry dan browser Node-RED.

3.1.3 Perancangan Skematik

cara kerja dari perancangan ini terdapat pada blok diagram keseluruhan. Pada perancangan simulasi ini tidak menggunakan *wire* atau kabel dalam perancangan melainkan digunakan node sebagai penghubung atau jumper.



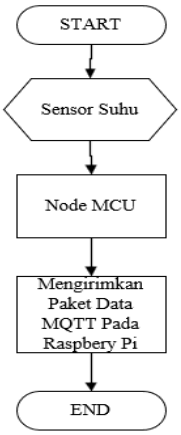
Gambar 8 Desain Perancangan Skematik MQTT

3.2 Perancangan Algoritma *Flowchart* Sensor

Terdapat tiga perancangan diagram alir untuk sensor yaitu sensor deteksi suhu(DHT-11), sensor deteksi kebocoran gas(MQ-2), dan sensor deteksi kebakaran (KY-026).

3.2.1 Perancangan *Flowchart* Sensor Suhu

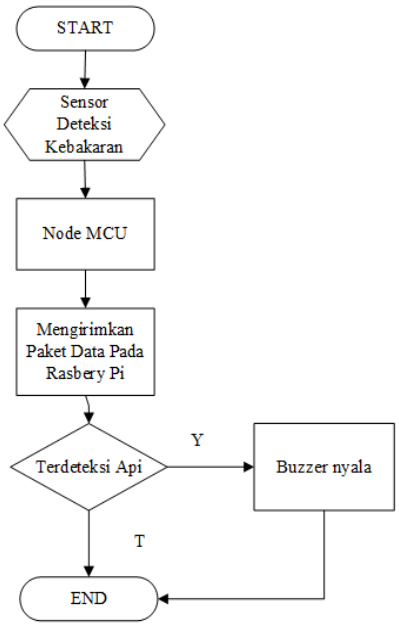
Pada flowchart untuk bagian sensor suhu, ketika sensor ini sudah terhubung dengan node MCU, dimana sebelumnya akan dimasukan kode program pada modul Node MCUESP8266 selanjutnya, sensor suhu akan memberikan informasi atau data yang nantinya akan dikirimkan kepada raspbery pi dan ditampilkan pada *browser* Node-RED melalui protokol MQTT. Outputnya ialah hasil dari suhu dan kelembaman dari ruangan. Kemudian setelah data diolah oleh raspbery pi nantinya akan ditampilkan hasil monitoring suhu dan kelembaman pada monitor UI(*User Interface*) *browser* Node-Red.



Gambar 9 Diagram Alir Sensor Suhu

3.2.2 Perancangan *Flowchart* Pendeteksi Api

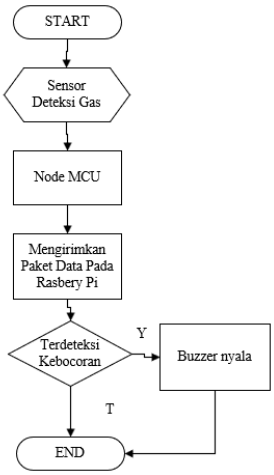
Pada flowchart untuk bagian sensor pendeteksi api, sensor ini nantinya akan terhubung dengan node MCUESP8266, dimana sebelumnya akan dimasukan kode program. Sensor akan memberikan informasi atau data yang nantinya akan diolah oleh raspbery pi sesuai dengan syarat dan kondisi tertentu. Misalnya, apabila nantinya sensor pendeteksi api mendeteksi bahwa adanya api maka, sensor selanjutnya akan mengirimkan data yang nantinya diolah oleh raspbery pi melalui protokol MQTT. Cara kerja sensor ini sendiri bukan mendeteksi berapa derajat suhu yang dikeluarkan oleh api untuk mendeteksi panas melainkan seberapa terangkah warna api yang dapat dihasilkan, semakin terang nyala api sensor infrared akan menangkap spektrum cahaya dan mengidentifikasi bahwa terdapat nyala api. Output yang dihasilkan ialah bilangan biner 1 dan 0. Apabila outputan dalam kondisi 1 berati terdapat nyala api yang terdeteksi dan sebaliknya outputan berlogik 0 maka tidak terdapat nyala api terdeteksi oleh sensor.



Gambar 10 Diagram Alir Sensor Ky-026

3.2.3 Perancangan *Flowchart* Sensor Deteksi Gas

Pada flowchart untuk bagian sensor pendeteksi kebocoran gas, sensor ini juga nantinya akan terhubung dengan raspbery pi, dimana sebelumnya akan dimasukan kode program, sensor akan memberikan informasi atau data yang nantinya akan diolah oleh raspbery pi sesuai dengan syarat dan kondisi tertentu. Misalnya, apabila nantinya sensor mendeteksi bahwa adanya kebocoran gas >400 ppm (*part per milion*) maka, sensor selanjutnya akan mengirimkan data dengan protokol MQTT mosquito kepada raspbery pi yang ditampilkan pada browser Node-RED. Setelah itu raspbery pi akan mengkonfirmasi bahwa adanya kebocoran gas yang terdeteksi. Output yang dihasilkan ialah besaran parameter satuan gas yaitu ppm(*part permilion*).



Gambar 11 Diagram Alir Sensor MQ-2

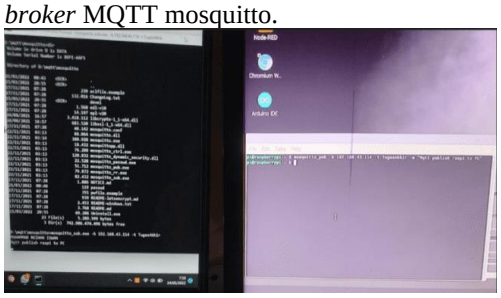
4. HASIL DAN PEMBAHASAN

Setelah dilakukan perancangan baik itu *hardware* ataupun *software* didalam pengimplementasian TLS sebagai metode keamanan protokol jaringan untuk MQTT, didapat hasil-hasil serta pembahasan sebagai berikut:

Pada gambar 12 dibawah merupakan sebuah realisasi *hardware* dari perancangan MQTT. Dibuat sebuah maket atau denah rumah seperti pada diagram ilustrasi sebagai konsep MQTT yang dibangun dengan tema "*smart home"*.

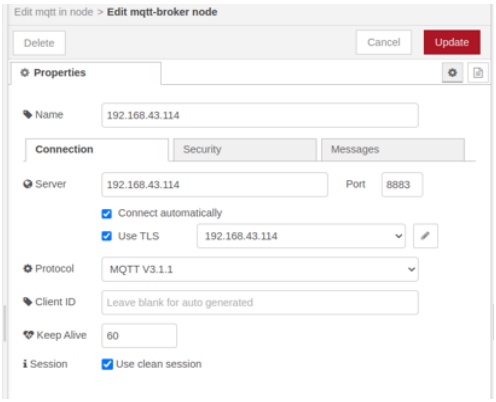
Gambar 12 Realisasi Pada *Hardware*

Pada gambar 13 ialah parameter hasil uji dari komunikasi MQTT antar *device*. Dimana, dalam pengujiannya digunakan Cmd(*Command Prompt*) pada PC dan terminal pada raspberry pi untuk menguji *broker* MQTT mosquitto.



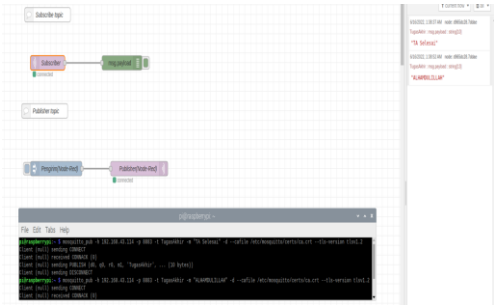
Gambar 13 Uji Komunikasi Antar *Device*

Pada gambar 14 adalah parameter hasil penerapan implementasi protokol TLS(*Transport Layer Security*) untuk MQTT yang diimplementasikan pada website Node-RED.



Gambar 14 Implementasi Protokol Keamanan TLS

Pada gambar dibawah yaitu gambar 15 adalah hasil parameter uji komunikasi pengiriman pesan baik dari *publisher* terminal raspberry pi menuju *subscriber* website Node-RED ataupun sebaliknya. Pada hasil pengujian dapat dinyatakan bahwa TLS berhasil diimplementasikan untuk mengenkripsi pesan atau *message* dalam protokol MQTT melalui browser Node-RED.



Gambar 15 Uji Komunikasi TLS Pada MQTT Melalui Browser Node-RED

Pada tabel dibawah menunjukan hasil parameter pengujian untuk penerapan TLS dalam protokol MQTT.

Tabel 1 Hasil Uji Parameter Pada MQTT

Komunikasi MQTT	**Implementasi TLS**	
	Bisa	**Tidak**
MQTT Komunikasi Terminal Localhost Raspberry Pi	√	
MQTT Komunikasi antar *device* PC-Raspberry Pi		√
MQTT Komunikasi Pesan via Node-RED	√	
MQTT Komunikasi IoT via Node-RED		√

Didalam pengujian yang telah dilakukan untuk menguji parameter pengimplementasian TLS sebagai metode protokol pengamanan tambahan didalam MQTT didapatkan bahwasannya untuk komunikasi protokol MQTT pada terminal localhost raspberry pi TLS(*Transport Layer Security*) bisa diimplementasikan kemudian sama halnya dengan komunikasi MQTT untuk pengiriman pesan melalui *browser* Node-RED. Akan tetapi untuk komunikasi pada MQTT antar *device* dan komunikasi yang dilakukan MQTT dengan integrasi IoT melalui browser Node-RED tidak dapat diimplementasikan.

Tabel 2 Hasil Uji Parameter QoS MQTT

Parameter MQTT	**Delay (ms)**	**PacketLoss(%)**
MQTT terminal Raspi *publisher* – NodeRed *subscriber*	0.000051 ms	0%
MQTT terminal Raspi *subscriber* – NodeRed *publisher*	0.00030 ms	0%
MQTT Sensor Suhu DHT-11	0.00024 ms	0.3%
MQTT Sensor Deteksi Kebocoran Gas MQ-2	0.00023 ms	0.1%
MQTT Sensor Deteksi Api KY-026	0.000083ms	0.2%

Tabel diatas merupakan hasil uji QoS(*Quality Of Service*) dari protokol MQTT. Terlihat dari hasil tabel diatas bahwasannya protokol MQTT memiliki delay komunikasi sangat rendah yang menandakan protokol ini sangat baik untuk digunakan dan pada *packet loss* atau data yang hilang disetiap pengiriman saat dilakukan komunikasi juga cenderung terbilang rendah yang dapat dinyatakan sangat baik.

5. KESIMPULAN DAN SARAN

Dari hasil dan pembahasan implementasi TLS sebagai metode dan usaha untuk keamanan tambahan pada protokol jaringan MQTT dapat ditarik kesimpulan diantaranya, untuk implementasi TLS sebagai metode keamanan pada MQTT berbasis raspberrypi ini berhasil diimplementasikan dan dapat mengenkripsi pengiriman pesan baik secara terminal localhost ataupun melalui browser NODE-RED. Akan tetapi, karena pada perangkat IoT memiliki sumber daya terbatas maksudnya

penulisan program terkait dengan RAM harus dibawah 4MB, sehingga protokol TLS ini tidak direkomendasikan untuk melakukan pengenkripsian dengan integrasi IoT.

Kemudian Pada protokol MQTT yang telah dilihat hasil uji QoS (*Quality Of Service*) ini dapat diketahui bahwasannya, protokol MQTT ini sangat baik dan dapat diimplementasikan serta dikembangkan untuk digunakan sebagai inovasi kemajuan masa yang akan mendatang dikarenakan, dari hasil pengujian *Quality Of Service* pada MQTT memiliki *delay* sangat rendah yang menandakan bahwasannya kemampuan protokol ini dalam mengirimkan data dapat dikatakan sangat cepat. Adapun terkait parameter dari *packet loss* untuk QoS protokol jaringan MQTT didapat hasil dari pengujian data menyatakan, dalam mengirimkan paket data berupa pesan persentase keberhasilannya mencapai 100% atau tidak ada paket data yang gagal dikirimkan. Namun, untuk MQTT yang terintegrasi dengan sensor IoT sendiri memiliki presentase dikisaran 99%.

Untuk saran didalam pengembangan MQTT yang diimplementasikan dengan menggunakan TLS(*Transport Layer Security*) masih diperlukan pengkajian lebih dalam dan seksama, terutama terkait refrensi-refrensi mengenai jenis metode keamanan apa saja yang lebih baik bisa diimplementasikan pada protokol jaringan MQTT.

6. DAFTAR PUSTAKA

[1] Y. Nurrohman, “Implementasi Protokol Message Queuing Telemetry Transport Pada Sistem Home Automation Studi Kasus Di Pt. Lskk Bandung,” Feb. 2019, Diakses: Jan. 20, 2022. [Online]. Available: http://elibrary.unikom.ac.id.

[2] “Implementasi Protokol MQTT Pada Monitoring Suhu Dan Ketersediaan Pakan Ikan Pada Akuarium | Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer.” https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/2381 (Diakses Jan. 21, 2022).

[3] “Blockchain-based identity and authentication scheme for MQTT protocol | 2021 The 3rd International Conference on Blockchain Technology.” https://dl.acm.org/doi/10.1145/3460537.3460549 (D. 21, 2022).

[4] M. Arman, “Rancang Bangun Pengamanan FTP Server dengan Menggunakan Secure Sockets Layer,” *J. Integr.*, vol. 9, no. 1, pp. 16–23, Apr. 2017, doi: 10.30871/JI.V9I1.272.

[5] “Implementation of SSL/TLS-based security mechanisms in e-commerce and e-mail applications using Java | Fowdur | Journal of Electrical Engineering, Electronics, Control and Computer Science.” https://jeeeccs.net/index.php/journal/article/view/91 (Diakses Feb. 03, 2022).

[6] N. Ntuli, A. Abu-Mahfouz, “A Simple Security Architecture for Smart Water Management System,” Procedia Computer Science, Volume 83, 2016, pp. 1164-1169.

[7] “Transport Layer Security (TLS) | hestanto personal website.” https://www.hestanto.web.id/transport-layer-security/ (DiaksesJan. 21, 2022).

[8] X. Wang, J. Zhang, E. Schooler, and M. Ion, “Performance evaluation of Attribute-Based Encryption: Toward data privacy in the IoT,” in Communications (ICC), 2014 IEEE International Conference on, June 2014, pp. 725–730.

[9] “Raspberry Pi (Definisi, Fungsi, Jenis, Spesifikasi dan Pemrograman) - KajianPustaka.com.” https://www.kajianpustaka.com/2020/12/Raspberry-Pi.html (Diakses Mar. 20, 2022)

[10] A. Niruntasukrat, C. Issariyapat, P. Pongpaibool, K. Meesublak, P. Aiumsupucgul and A. Panya, "Authorization mechanism for MQTT-based Internet of Things," 2016 IEEE International Conference on Communications Workshops (ICC), Kuala Lumpur, 2016, pp. 290-295..

[12] M. Singh, M. A. Rajan, V. L. Shivraj and P. Balamuralidhar, "Secure MQTT for Internet of Things (IoT)," Communication Systems and Network Technologies (CSNT), 2015 Fifth International Conference on, Gwalior, 2015, pp. 746-751.

[13] M. A. Iqbal and M. Bayoumi, "Secure End-to-End key establishment protocol for resource-constrained healthcare sensors in the context of IoT," 2017 International Conference on High Performance Computing & Simulation (HPCS), Innsbruck, 2017, pp. 523-530.