IoT Gateway Menggunakan Protokol MQTT pada Perangkat Kendali Berbasis Modbus-RTU

Imadudin Harjanto[1)]
[1)]Fakultas Teknik, Universitas PGRI Semarang,
Jl. Dr. Cipto – Lontar No. 1 Semarang; Telp.024-8451279.
Email: iharjanto@gmail.com

Abstrak

Salah satu tujuan dari IoT adalah bagaimana menghubungkan perangkat yang ada kedalam jaringan infrastruktur teknlogi dan informasi. Meskipun produk-produk baru sudah mendukung untuk integrasi ke dalam IoT, namun masih banyak perangkat eksisting yang belum mendukung untuk itu. Salah satunya adalah perangkat kendali Modbus RTU. Salah satunya adalah Autonic TK 4S pengontrol suhu. Alat ini masih menggunakan komunikasi serial sehingga sulit untuk integrasi ke dalam jaringan IoT. Penelitian ini bertujuan untuk merancang gerbang IoT yang memungkinkan perangkat yang identik, berbasis komunikasi ModbusRTU agar dapat berintegrasi dengan jaringan IoT dengan tetap memperhatikan prinsip fleksibilitas, keamanan data, kecepatan data dan bandwidth. Dari hasil percobaan dihasilkan bahwa integrase perangkat Modbus RTU dapat diimplementasikan dengan menambahkan gerbang IoT berupa raspberry pi yang berfungsi ganda sebagai Modbus Master dan MQTT Client. Komunikasi raspberry ke perangkat yang terhubung ke sensor dan actuator masih tetap menggunakan protokol Modbus, dan komunikasi ke internet dilakukan dengan protokol MQTT dengan mode publish/subscribe. Hasil percobaan menunjukkan dapat diterapkannya mode keamanan minimal pada scenario ini dengan menambahkan username dan password. Sedangkan uji unjuk kerja menunjukkan hasil komunikasi dengan latency yang masih rendah di bawah 0.264 detik untuk keseluruhan komunikasi dari perangkat Modbus-MQTT broker – klien. Sedangkan karakter dari mqtt dan Modbus itu sendiri yang bersifat lightweight dengan data rate rendah masih bisa dipertahankan dengan konsumsi bandwidth maksimal sebesar 9Kbps.

Kata kunci: IoT, Modbus RTU, MQTT, system kendali, SCADA.

Abstract

One of the main goal of IoT implementation is to integrate existing device to the global information and communication infrastructure. Nowadays, most of new product have been designated to be IoT ready with, some feature to easily integrated to the internet world. The challenging is how to integrate old existing devices which doesn't support new communication protocol. One of example is Autonics TK4S, a temperature monitoring controller based on Modbus RTU serial line communication which is very usual in industry. This paper discussed the scenario how to implement IoT to the device with Modbus RTU communication protocol to integrate with global internet while still maintaining the Modbus and MQTT character: lightweight and flexibility. Next thing to discuss is the security problem in Modbus. After some examination, we get result that security can be implemented by adding authentication. Meanwhile, the lightweight and flexibility still can be maintained. In our scenario, latency of the system is still low at less than 0,3 s for Modbus-MQTT broker-client communication and less than 0,01 s for the MQTT communication alone. Data rate is still can be maintained at less than 9kbps.

Keywords: *IoT, Modbus RTU, MQTT, control system, SCADA, lightweight*

1. PENDAHULUAN

Teknologi *Internet of Things* (IoT) telah mengalami perkembangan yang luar biasa didukung dengan perkembangan infrastruktur informasi dan teknologi. IoT memungkinkan perangkat apapun dapat terhubung ke dalam jaringan dan dapat berinteraksi untuk keperluan monitoring dan administrasi perangkat. Hal ini tidak terkecuali dalam dunia automasi industri yang mana banyak perangkat kontrol yang tersebar di berbagai lokasi industri. Tidak dipungkiri, protokol komunikasi antar perangkat yang paling banyak digunakan saat ini adalah protokol Modbus. Protokol Modbus versi terbaru mampu mendukung komunikasi berbasis TCP, sehingga mudah terkoneksi dengan perangkat kontrol maupun monitor lain yang berada di luar jaringan. Namun banyak tidak banyak juga industri yang masih memanfaatkan protokol modbus berbasisi komunikasi serial, yaitu Modbus RTU. Salah satu contoh implementasi Modbus RTU adalah untuk komunikasi pada sistem PLC dan SCADA (Desai, et al., 2017).

Pada perangkat kontrol yang masih berbasis komunikasi modbus RTU, permasalahanya adalah

p-ISSN 2460-9986
e-ISSN 2476-9436

komunikasi masih menggunakan komunikasi serial dengan protokol RS485. Komunikasi serial memiliki keterbatasan dalam hal interoperabilitas, tantangan dalam menghubungkan dengan dunia luar. Pengembangan dari peUntuk komunikasi berbasis Modbus-TCP, beberapa peneliti sebelumnya telah mengeimplementasikan protokol Modbus dan merancang gateway ke dalam jaringan IoT (Suryawanshi, et al., 2019).

Abilovani dalam penelitiannya menemukan bahwa implementasi protokol MQTT dalam monitoring jaringan memiliki efisiensi yang lebih baik dengan parameter bandwidth yang rendah dan lattency yang tinggi (Abilovani, et al., 2018). Paket pada protokol MQTT memiliki ukuran yang jauh lebih rendan dibandingkan pada UDP untuk SNMP. Dalam penelitian lain oleh Saputra (Saputra, et al., 2017), MQTT yang digunakan dalam komunikasi WAN memiliki packet loss 0% dan akurasi pengiriman 100%.

Selain faktor fleksibilitas, kecepatan data, bandwidth dan lattency yang berkaitan erat dengan infrastruktur jaringan komunikasi, faktor keamanan juga tidak kalah penting dalam memilih protokol apa yang sebaiknya digunakan dalam komunikasi perangkat kendali. Penelitian tentang keamanan telah dilakukan oleh Andy dengan penemuan bahwa penggunaan protokol MQTT dapat dikatakan cukup aman dengan catatan penerapan skenario keamanan melalui implementasi TLS (Andy, et al., 2017). Meskipun penerapan TLS dapat meningkatkan penggunaan bandwidth. Hal lain yang dapat dilakukan misalnya dengan menerapkan penerapan penyandian RSA (Ádámkó, et al., 2017) yang mana dengan skenario ini penghematan bandwidth masih tetap dipertahankan.

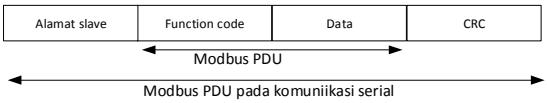
Dengan mempertimbangkan beberapa faktor diatas, faktor konektifitas, bandwidth, kecepatan data, lattency dan faktor keamanan, skenario implementasi gateway IoT pada perangkat kendali berbasis Modbus RTU dengan komunikasi serial bisa diimplementasikan melalui penggunaan protokol komunikasi pesan MQTT.

2. LANDASAN TEORI

2.1. Protokol ModBus

Protokol Modbus merupakan struktur pesan pada lapisan aplikasi yang banyak digunakan untuk komunikasi *master-slave* antar perangkat kendali (Modbus.Org, 2012). Pesan Modbus dikirim dari *Master* ke pada *Slave* yang memuat alamat *slave*, perintah, data dan *checksum*. Karena hanya berbentuk pesan, maka protocol ini tidak bergantung pada lapisan fisik yang digunakan, namun pada umumnya protocol ini diimplementasikan pada jaringan serial menggunakan protocol RS 232, RS 485 dan atau jaringan IP menggunakan protocol TCP pada versi Modbus-TCP.

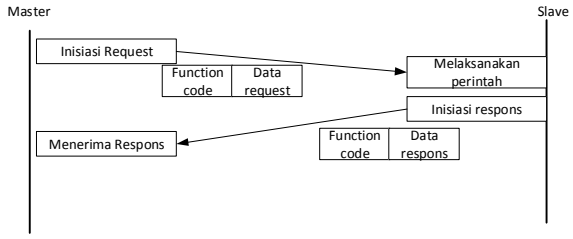
Protokol Modbus mendefinisikan *protocol data unit* (PDU) terdiri atas alamat, function code, data dan checksum. Pesan dapat dibedakan menjadi dua yaitu Request dan Respons. Request berisi function code yang memerintahkan slave dengan alamat yang dituju untuk melakukan aksi atas perintah yang diberikan, dapat berupa perintah untuk membaca atau menulis data pada register yang dituju. Sedangkan Respons merupakan dari slave yang merupakan tanggapan atas perintah yang diberikan master.



Gambar 1 Format data PDU prtokol Modbus

Alamat slave merupakan identifikasi slave berupa rentang desimal antara 0 – 247. Master mengirim pesan dengan menambahkan alamat slave yang dituju pada awal pesan, sedangkan slave memberikan alamat diri sendiri sebagai respon agar master mengetahui slave mana yang merespon. *Function code* mengindikasikan perintah apa yang diminta oleh master agar dikerjakan oleh slave. *Function code* dapat diikuti dengan data perintah dari master atau data status pada respons slave. CRC merupakan pendeteksi kesalahan yang merupakan perhitungan *redundancy check* pada pesan yang dikirim

Data pada aplikasi Modbus dibangun melalui transaksi pesan Modbus yang ditunjukkan pada gambar 2.

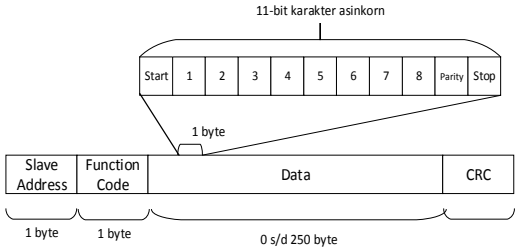


Gambar 2 Transaksi pesan pada protokol Modbus

Protokol Modbus memungkinkan transmisi serial dengan dua mode yaitu RTU dan ASCII (Modbus.Org, 2006). Pada penelitian ini, digunakan mode RTU. Format Modbus RTU mendefinisikan bagaimana informasi dikemas dalam bentuk pesan dan bagaimana dikodekan. Format transmisi ini harus sama untuk semua perangkat agar komunikasi dapat berjalan. Pada komunikasi transmisi serial dengan mode RTU, masing-masing pesan disusun oleh 8-bit byte yang terdiri dari 2 karakter dalam format hexadecimal masing-masing 4 bit. Selain pesan 8 bits,

p-ISSN 2460-9986
e-ISSN 2476-9436

masih terdapat 1 start bit, 1 bit paritas dan 1 stop bit, sehingga dalam 1 byte RTU, tersusun atas 11 bit karakter.



Gambar 3 Format data frame Modbus RTU

Tabel 1 Deskripsi dan spesifikasi fungsi kode modbus

Function Code	Deskripsi	spesifikasi data	
		request	respons
01 READ COIL STATUS	membaca status coil diskret pada slave	awal register coil dan jumlah bit yang dibaca	tanggapan status coil per bit
02 READ INPUT STATUS	membaca status input diskret pada slave	awal register input dan jumlah bit yang akan dibaca	tanggapan status input per bit
03 READ HOLDING REGISTERS	membaca konten binari register holding pada slave	awal register holding dan jumlah register yang dibaca	tanggapan status register holding berupa 2 byte data per register
04 READ INPUT REGISTERS	membaca konten binari register input pada slave	awal register input dan jumlah register yang dibaca	tanggapan status register input berupa 2 byte data per register
05 WRITE SINGLE COIL	menulis coil tunggal ON/OFF	referensi coil yang akan ditulis dan status ON/OFF berupa konstanta dengan nilai hexa 0x00FF untuk on dan 0x0000 untuk off	respons normal adalah echo dari pesan yang di request setelah coil berhasil ditulis
06 WRITE SINGLE REGISTER	menulis sebuah nilai pada register tunggal	referensi register tunggal yang akan ditulis dan data register	respons normal adalah echo dari pesan yang di request setelah register berhasil ditulis
15 WRITE MULTIPLE COILS	menulis data coils ON/OFF dalam bentuk sekuens	referensi atas coil yang akan ditulis dan nilai coil 1 untuk ON dan 0 untuk OFF	respons normal adalah alamat slave, function code, dan jumlah coil yang berhasil ditulis
16 WRITE MULTIPLE REGISTERS	menulis register dalam bentuk sekuens	referensi atas register yang akan ditulis dengan data register 2 bytes per register	respons normal adalah alamat slave, function code, dan jumlah register yang berhasil ditulis

2.2. Protokol MQTT

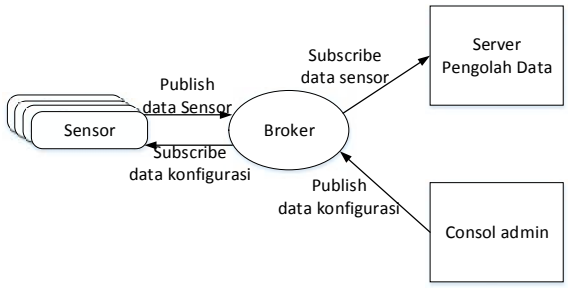
Protokol MQTT pertama kali di kembangkan oleh IBM pada akhir 1990-an. Pada awalnya protokol ini dikembangkan untuk menghubungkan sensor pada jalur pipa minyak dengan satelit. Protokol ini digunakan untuk pertukaran pesan secara asinkron, yang mana antar pengirim dan penerima terpisahkan oleh jarak dan waktu, yaitu tidak harus berkomunikasi secara sinkron.

Protokol MQTT merupakan protocol yang ringan (*lightweight*), artinya dapat diemplementasikan pada keadaan terbatas pada sisi perangkat maupun pada keterbatasan jaringan karena keterbatasan *latency/bandwidth.* Disisi lain, protocol MQTT bersifat fleksibel, yaitu dapat diterapkan pada beragam

p-ISSN 2460-9986
e-ISSN 2476-9436

skenario baik dalam konfigurasi perangkat maupun layanan yang diberikan (Yuan, 2017).

Protokol MQTT menggunakan model komunikasi *publish* dan *subscribe*. Pengguna dibedakan menjadi dua macam yaitu *message broker* dan sejumlah klien. Broker adalah server yang menampung semua pesan dari semua klien dan kemudian merutekan pesan ke klien yang dituju. Klien adalah semua benda yang dapat berinteraksi dengan broker untuk mengirimkan dana tau menerima pesan. Klien tersebut dapat berupa sensor yang mengirimkan data variable terukur, maupun pusat data yang mengolah data secara keseluruhan.

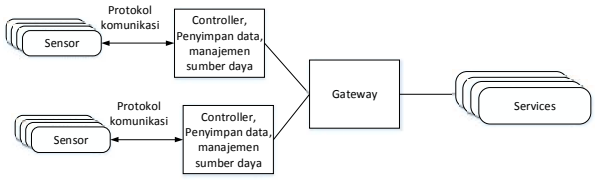


Gambar 4 Arsitektur jaringan protokol MQTT

2.3. Internet of Things

Menurut definisi ITU, *Internet of Things* (IoT) dapat dipandang sebagai sebuah infrastruktur global bagi masyarakat informasi, yang memungkinkan adanya layanan mutakhir dengan menghubungkan berbagai benda (baik secara fisik maupun virtual) berdasarkan pada perangkat informasi teknologi dan komunikasi (ICT) yang telah ada (ITU-T, 2012). Benda (*things*) yang menjadi obyek dalam IoT adalah entitas fisik dan atau informasi virtual yang dapat diidentifikasi dan diintegrasikan kedalam jaringan komunikasi secara umum. Informasi yang diolah dapat berupa informasi dinamis maupun statis. Benda secara fisik dapat berupa obyek yang dapat dirasakan, dan terhubung ke dalam jaringan, misalnya robot, perangkat rumahtangga, atau sensor. Sedangkan informasi virtual adalah informasi yang dapat disimpan, diproses dan diakses, misalnya informasi pada konten multimedia, atau aplikasi peerangkat lunak.

Secara umum, arsitektur IoT terdiri dari sensor, *controller*, *gateway* dan penyedia layanan yang siap digunakan oleh entitas lain.



Gambar 5 Arsitektur jaringan IoT

Karakter mendasar pada system IoT adalah sebagai berikut (Zennaro, 2017):

a) **Interkonektifitas** : apapun dapat dihubungkan secara global melalui jaringan infrastruktur komunikasi dan informasi

b) **Heterogenitas** : perangkat IoT dapat terdiri dari platform jaringan maupun perangkat keras yang heterogen, namun harus tetap dapat berinteraksi satu sama lain.

c) **Perubahan dinamis**: status keadaan perangkat yang terhubung ke jaringa IoT dapat berubah secara dinamis dari keadaan Idle, Active, Dormant ataupun terhubung/tidak terhubung. Sehingga perangkat terhubung dapat berubah secara dinamis.

d) **Skalabilitas**: karena siftanya yang dinamis, system IoT harus bisa beradaptasi terhadap skala ukuran jaringan dari yang terkecil dan mampu melayani perkembangan jumlah perangkat, karena tren saat ini komunikasi data di masa mendatang akan lebih banyak di dominasi oleh perangkat dibandingkan oleh informasi yang di bangkitkan oleh manusia.

Dalam perancangan system Io Tada beberapa aspek kunci yang harus diperhatikan antara lain:

a) **Jangkauan**, seberapa besar jaringan yang akan dibangun

b) **Kecepatan Data**, seberapa besar bandwidth yang dibutuhkan untuk mengakomodasi perubahan data pada obyek.

c) **Daya**, bagaimana sumber daya/baterai pada obyek sensor.

d) **Keamanan**, jika melibatkan data dengan informasi bersifat rahasia, tentu harus memperhatikan aspek keamanan data.

3. METODOLOGI PENELITIAN

3.1. Perumusan Masalah

Protokol Modbus-RTU merupakan modbus yang paling banyak digunakan dalam perangkat kendali. Salah satu produk yang akan digunakan dalam penelitian ini adalah Temperatur Controller seri TK. Pada Modbus berbasis TCP, interkoneksi dengan jaringan eksternal tidak terlalu rumit, karena

p-ISSN 2460-9986
e-ISSN 2476-9436

komunikasi sudah berbasis pada protokol IP. Untuk dapat mengintegrasikan perangkat Modbus RTU ke dalam jaringan IoT, diperlukan perangkat tambahan sebagai gateway dan protokol komunikasi. Dengan pemaparan pada bagian pengantar diatas, penelitian ini bertujuan untuk menentukan konfigurasi yang tepat dan mengukur kinerja sistem yang dibangun

3.2. Perancangan sistem.

Tujaun dari penelitian ini adalah mengimplementasikan IoT gateway berbasis protokol komunikasi MQTT agar perangkat kendali dapat terhubung ke jaringan internet secara aman dan tetap mempertahankan kerakteristik Modbus yaitu format data sederhana dan fleksibel. Pemilihan protokol MQTT didasarkan pada karakteristik yang hampir sama dengan Modbus yaitu fleksibilitas dan ringan. Dengan skenario ini, diharapkanperangkat Modbus tidak perlu dilakukan upgrade baik fisik perangkat maupun software. Sebagai gantinya, akan ditambahkan perangkat yang berfungsi sebagai gateway ke Internet dan penghubung antara protokol ModbusRTU dan MQTT.

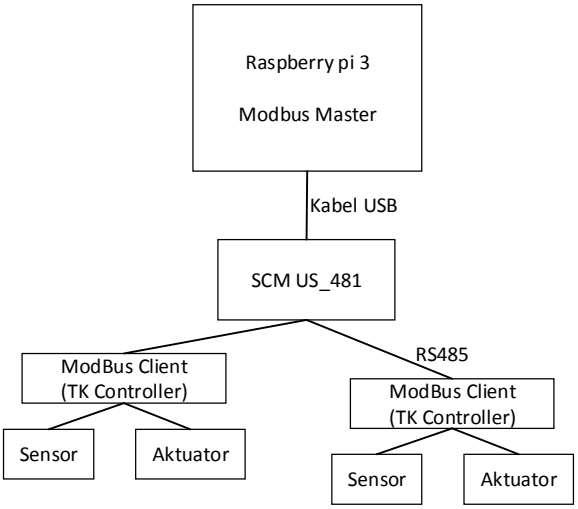
3.2.1. Analisa kebutuhan sistem

Sistem yang akan dibangun berusaha tetap mempertahankan kondisi eksisting, yaitu adanya pernagkat kendali berbasis ModbusRTU tanpa megganti perangkat namun dapat mengintegrasikannya ke dalam jaringan internet. Alat dan bahan yang diperlukan dalam perancangan ini adalah :

a. Perangkat kendali berbasis Modbus RTU, berupa TK 4S Temperatur Controllor Autonics.
b. Konverter USB to Serial SCM US481
c. Sensor temperature sebagai pembangkit data yang diamati.
d. Kabel RS485 untuk koneksi antara Perangkat Modbus dengan Konverter USB
e. Raspberry pi 3
f. MQTT broker server
g. Akses internet berupa access point Wifi dengan kecepatan max 10 Mbps
h. MQTT klient dapat berupa computer/laptop/atau Smartphone
i. Program python dengan library pendukungnya yaitu : pyserial, pymodbus, dan paho-mqtt
j. Software tcp dump pada raspberry pi untuk keperluan paket sniffer.
k. Software Wireshark untuk pembacaan dan analisa paket data hasil capture dari raspberry.

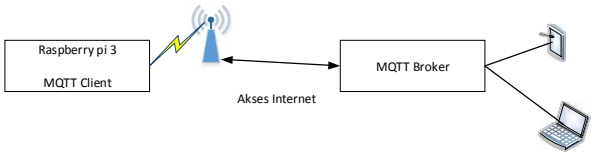
3.2.2. Perancangan arsitektur sistem

Implementasi sistem dapat dibagi menjadi dua bagian yaitu bagian jaringan komunikasi internal perangkat kendali dan jaringan eksternal yang akan digunakan oleh pengguna system dari luar jaringan. Jaringan internal terdiri dari sensor dan actuator, perangkat kendali Modbus RTU sebagai slave Modbus, dan raspberry py yang berlaku sebagai master Modbus



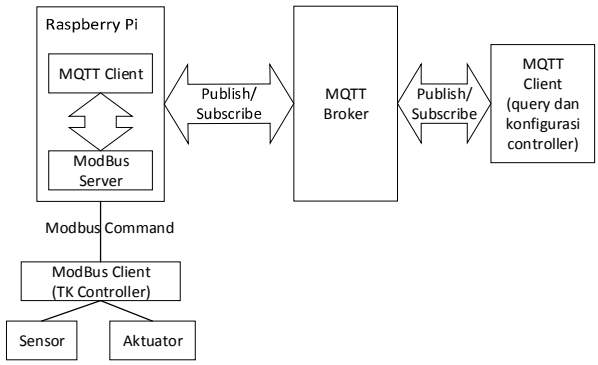
Gambar 6 Arsitektur jaringan fisik internal

Sedangkan Jaringan eksternal adalah penghubung antara Modbus server yang berfungsi sebagai pengumpul data internal yang dikirimkan ke server MQTT Broker. Kemudian MQTT broker akan melayani permintaan atau pengiriman data oleh klien di luar jaringan internal.



Gambar 7 Arsitektur jaringan fisik eksternal

Kedua jaringan fisik tersebut diatas kemudian diintegrasikan menggunakan protokol komunikasi Modbus RTU dan MQTT melalui program python yang ditanamkan dalam Raspberry Pi.



Gambar 8 Intergrasi Modbus RTU dan MQTT

p-ISSN 2460-9986
e-ISSN 2476-9436

3.2.3. Integrasi system dan perancangan program python

Berdasarkan rancangan arsitektur diatas, dibutuhkan perangkat lunak untuk integrasi jaringan fisik internal dan jaringan eksternal. Dalam hal ini perangkat lunak ditulis dalam bahasa pemrograman python. Dalam perangkat lunak tersebut, terdapat dua modul yaitu :

a) Modul Modbus.

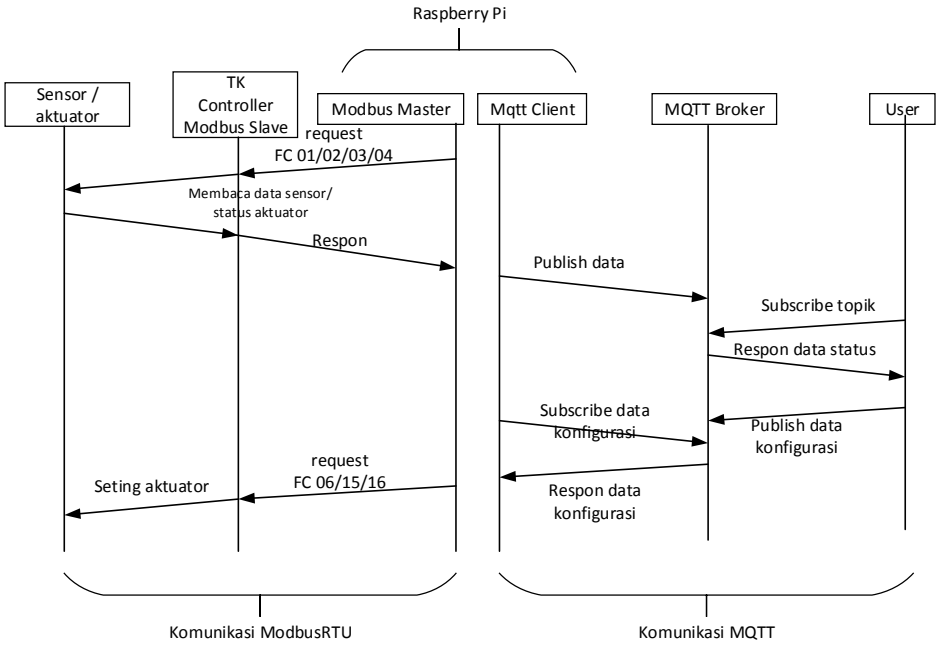
Controller TK 48S Autonics sebenarnya sudah memiliki perangkat lunak pembaca sensor dan antar muka ke pengguna. Namun Perangkat lunak tersebut belum dilengkapi dengan antarmuka ke jaringan eksternal. Perangkat lunak juga hanya dapat diinstal pada system operasi Windows yang tidak mungkin diterapkan pada raspberry. Untuk itu diperlukan pembaca parameter Modbus yang dapat ditulis dengan menggunakan Bahasa python dengan memanfaatkan library pymodbus.

Fungsi yang dapat dijalankan dalam modul ini adalah kode fungsi yang sejalan dengan protokol Modbus yaitu membaca/menulis input diskret dan membaca/menulis input register.

b) Modul MQTT

Setelah parameter pada perangkat Modbus dibaca dengan menggunakan modul Modbus, data tersebut dikirimkan oleh modul MQTT ke server MQTT Broker.

Fungsi yang dijalankan oleh modul ini adalah mengirimkan data dari perangkat Modbus ke mqtt broker melalui perintah publish dan pembacaan konfigurasi data yang dikirim oleh administrator dari jaringan eksternal melalui perintah subscribe. Skenario dari komunikasi data digambarkan seperti gambar 9.



Gambar 9 Prosedur komunikasi data

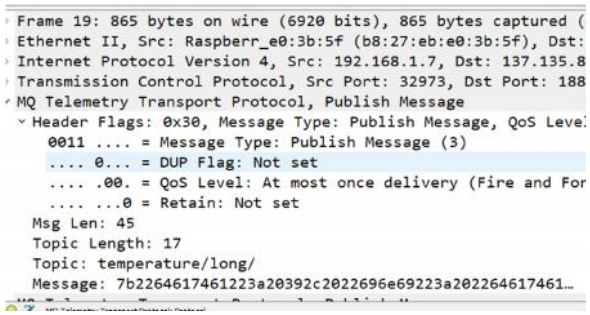
4. HASIL DAN PEMBAHASAN

Implementasi dari skenario diatas telah berhasil dilakukan dengan baik. Secara keseluruhan koneksi peralatan dapat di lakukan. Mode komunikasi data dilakukan dengan 2 arah yaitu mode baca dan mode setting. Keduanya berhasil dilakukan, yaitu user dapat membaca data status pada sensor/aktuator melalui perantara server MQTT broker dan konfigurasi jaringan juga berhasil dilakukan sesuai skenario pada gambar 9.

4.1. Keamanan Jaringan

Dengan format data yang sederhana, packet data MQTT bersifat terbuka. Data dapat dilihat dengan transparan ketika dibuka melalui packet sniffer. Hal ini tentu akan memberikan celah keamanan bagi pihak ketiga yang dapat membaca atau memalsukan data yang dikirim.

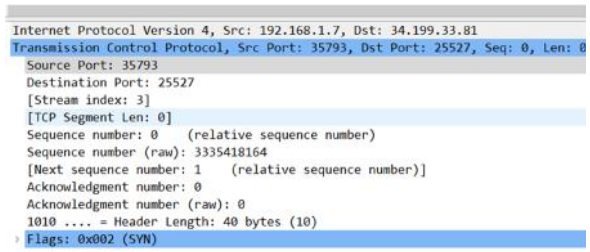
p-ISSN 2460-9986
e-ISSN 2476-9436



Gambar 10. Tangkapan paket data MQTT tanpa keamanan

Pada gambar 9 di atas terlihat data payload MQTT yang dikirim terlihat dengan jelas, terdapat topic dan pesan datanya.

Pada percobaan ini dicoba dengan menambahkan fitur authentikasi dengan merubah port default dari 1883 menjadi port 25527 dan memberikan username dan password pada protokol MQTT. Hasilnya paket data menjadi lebih aman dan tidak dapat terbaca ketika dilakukan penangkapan menggunakan paket sniffer. Dengan penambahan keamanan, pada tangkapan paket data MQTT, payload menjadi tidak terlihat, seperti ditunjukkan oleh gambar 10 berikut.



Gambar 11. Tangkapan paket data MQTT yang dilengkapi dengan fitur keamanan

Agar kemanan lebih terjamin, terdapat pilihan fitur keamanan pada protokol MQTT antara lain enkripsi menggunakan protokol SSL-TLS.

4.2. Uji Unjuk kerja jaringan

Dalam pengujian unjuk kerja jaringan berdasarkan scenario di atas, pada penelitian ini dilakukan pengujian dengan skenario berikut:

a. Pengukuran unjuk kerja MQTT melalui pengiriman perintah Publish dan subscribe

b. Pengukuran unjuk kerja keseluruhan melalui pengiriman perintah Modbus dan MQTT secara berurutan.

Hasil dari pengukuran ditunjukkan pada Tabel 2.

Tabel 2. Hasil Pengukuran latency dan throughput

skenario	Modbus Payload (byte)	Lattency (s)	Throughput Bps
Modbus query + MQTT Publish	8	0,262874	513,027
Modbus query + MQTT Publish	1	0,284491	148,1039
MQTT Subscribe	-	0,010563	180,2763
MQTT Publish	-	0,010072	9320,601

5. SIMPULAN

Upaya untuk mengimplementasikan IoT pada perangkat kendali di industri yang berbasis ModbusRTU dapat dilakukan dengan menambahkan gerbang IoT dengan menggunakan protokol yang dapat diaplikasikan menggunakan raspberry pi 3 dan perangkat lunak yang dibangun menggunakan python.

Dengan integrase ModbusRTU dan MQTT ini, dapat meningkatkan keamanan karena protokol Modbus RTU sendiri pada dasarnya memiliki resiko keamanan yang tinggi, karena protokol Modbus RTU merupakan salah satu protokol yang terbuka, siapapun dapat mengakses dan memberikan perintah melalui function code. Dengan adanya integrase ke MQTT, maka fitur keamanan pada MQTT dapat diadopsi pada system. Pada penelitian selanjutnya masih bisa dikembangkan dengan menerapkan fitur keamanan yang lebih handal misalnya dengan enkripsi menggunakan protokol SSL-TLS

Dari hasil percobaan juga terlihat bahwa latency dibawah 0,3 detik dan datarate masih dibawah 10kBps, masih cukup rendah dan ringan untuk diimplementasikan pada infrasktruktur minimal.

6. DAFTAR PUSTAKA

Abilovani Zavero Brillianata, Yahya Widhi dan Bakhtiar Andri Fariz Implementasi Protokol MQTT Untuk Sistem Monitoring Perangkat IoT [Jurnal] // Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer. - 2018. - 12 : Vol. 2. - hal. 7521.

Ádámkó Éva, Jakabóczki Gábor dan Szemes Péter Tamás Proposal of a Secure Modbus RTU Communication with Adi Shamir's Secret Sharing Method [Jurnal] // INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS. - 2017. - 2 : Vol. 64. - hal. 107-114.

Andy Syaiful, Rahardjo Budi dan Hanindhito Bagus Attack Scenarios and Security Analysis of MQTT Communication Protocol in IoT System

p-ISSN 2460-9986
e-ISSN 2476-9436

[Konferensi] // EECSI 2017. - Yogyakarta, Indonesia : [s.n.], 2017.

Desai Manan M [et al.] Modbus Control System using PLC and SCADA [Jurnal] // IJSTE - International Journal of Science Technology & Engineering. - 2017. - 9 : Vol. 3. - hal. 257-262.

ITU-T Overview of the Internet of things [Buku]. - [s.l.] : International Telecommunication Union, 2012.

Modbus.Org Modbus Application Protocol Specification [Buku]. - Massachusetts : Modbus.org, 2012.

Modbus.Org MODBUS over Serial Line Specification and Implementation Guide [Buku]. - [s.l.] : Modbus.Org, 2006.

Saputra Galih Yudha, Afrizal Ahimsa Denhas dan Mahfud Fakhris Khusnu Reza Penerapan Protokol MQTT pada Teknologi WAN (Studi Kasus Sistem Parkir Universitas Brawijaya) [Jurnal] // Jurnal Informatika Mulawarman. - 2017. - 2 : Vol. 12. - hal. 69.

Suryawanshi Shweta [et al.] IoT Gateway Design and Implementation for Modbus Protocol [Jurnal] // International Journal of Engineering Applied Sciences and Technology. - 2019. - 6 : Vol. 4. - hal. 104-107.

Yuan Michael Getting to know MQTT [Artikel] // IBM Developer Article. - 12 May 2017.

Zennaro Marco Introduction to the Internet of Things [Laporan]. - [s.l.] : Internationa Telecommunication Union, 2017.

.