Journal of Technology and Informatics (JoTI)
Vol. 2, No.1, Bulan Oktober Tahun 2020
E-ISSN : 2686-6102
P-ISSN : 2721-4842

Implementasi dan Analisis Fitur Keamanan Protokol MQTT pada *Telehealthcare*

Dian Rachmadini[1] **Ira Puspasari**[2] **Jusak**[3]
Program Studi/Jurusan Teknik Komputer, Universitas Dinamika
Email: 16410200026@dinamika.ac.id[1], ira@dinamika.ac.id[2], jusak@dinamika.ac.id[3]

Abstrak: Protokol MQTT merupakan salah satu protokol IoT dengan konsep *publisher, subscriber, dan broker*. Penerapan protokol MQTT dengan fitur keamanan dapat menggunakan *Transport Layer Security* (TLS). Pada makalah ini, fitur keamanan TLS pada MQTT akan diimplementasikan untuk mengirim data sinyal EKG. Sinyal EKG berbeda setiap orang dan merupakan privasi bagi pasien, karena dari sinyal EKG ini juga dapat digunakan untuk melihat penyakit pasien tersebut. Proses transmisi pada *MQTT Security* (MQTTS) akan menggunakan file kunci yang telah dibuat oleh *broker*. File kunci ini akan diberikan kepada *client* supaya bisa melakukan komunikasi, mengirim dan menerima data yang telah terenkripsi. Enkripsi data dilakukan karena adanya file kunci ini sehingga proses transmisi lebih aman. Hasil analisis perhitungan selisih besar paket sebelum dan setelah pengiriman pada QoS 0 adalah 152,6458 byte dan 139,4504 byte. Sedangkan QoS 1 sebesar 99,7932 byte dan 115,5321 byte. Kedua QoS menunjukkan selisih yang cukup besar, tetapi disisi lain pegiriman data menjadi lebih aman. Pada pengujian waktu yang diperlukan untuk proses enkripsi, QoS 0 menghasilkan waktu rata-rata 0,7 ms, QoS 1 menunjukkan hasil lebih lama yaitu 9,6 ms dikarenakan penambahan sinyal kontrol pada QoS 1. Hasil uji integritas data dengan *cross-correlation* , QoS 0 dan QoS 1 menunjukkan nilai 1 pada lag ke-0 yang artinya data yang dikirim dan diterima tidak terdapat perubahan (sama).

Kata Kunci: Protokol MQTT, MQTT *Secure,* Sinyal EKG.

Abstract : *MQTT protocol is one of the widely used lightweight IoT protocols. To secure the data transmission, the protocol enables security features by employing transport layer security (TLS). In this paper, such MQTT security (MQTTS) will be implemented to transmit electrocardiogram (ECG) data. Indeed, ECG data contains a unique identification of a person, thus its security should be maintained properly. To achieve that goal, transmission process on the MQTTS utilizes key files that have been created by the broker. This key file will be transmitted to the client so that they can communicate, send and receive the encrypted data. In this study, we measured the effect of encryption on the packet length, computation time and data integrity by activating both QoS 0 and QoS 1. Examination results showed that packet sizes before and after encryption for QoS 0 were 152.6458 bytes and 139.4504 bytes, consecutively. On the other hand, packet sizes for QoS 1 were 99.7932 bytes before encryption and 115.5321 bytes after encryption. It showed that for both QoS, the packet size growing larger yet the data enjoyed more secure transmission. Evaluation on the encryption processing time showed an average time of 0.7 ms for QoS 0 and a longer processing time i.e., 9.6 ms for QoS 1. This is mainly due to the addition of control signals in QoS 1. Data integrity was tested using the cross-correlation function. The study showed that for both QoS 0 and QoS 1 indicated a cross-correlation of 1 at the 0th lag. It can be concluded that the sent and the received data are exactly the same.*

Keywords: *MQTT Protocol, MQTT Secure, Electrocardiogram (ECG)*

PENDAHULUAN

Konsep *Internet of Things* atau biasa dikenal dengan IoT saat ini sudah tidak asing lagi, berbagai bidang pekerjaan dan aktifitas manusia menjadi lebih ringan dan terbantu karena menerapkan konsep ini. Salah satu protokol yang digunakan untuk IoT adalah protokol *Message Queue Telemetry Transpor* (MQTT) [1]. Protokol MQTT memiliki konsep yang sederhana yang terdiri dari *publisher, broker*, dan *subscriber*. Ketiganya saling dihubungkan dengan topik yang menentukan si penerima data. *Subscriber* hanya akan menerima data dari *publisher* yang memiliki topik sama. Pembuatan topik tidak dibatasi dan dibuat di sisi *publisher.*

Pada tahun 2019, M. Reza Bintami [2] melakukan penelitian dengan menerapkan protokol MQTT untuk pengiriman sinyal *Heartrate.* Sinyal *Heartrate* merupakan bagian dari sinyal EKG, dimana pada penelitian tersebut implementasi protocol MQTT dilakukan dengan analisis pengiriman data memanfaatkan fitur QoS 0 dan QoS 1. Penelitian ini menunjukkan hasil bahwa penggunaan protokol MQTT pada

parameter *packet loss* dan *delay* tergolong sangat bagus, karena *packet loss* yang dihasilkan kurang dari 1% dan rata-rata *delay* 20,21 ms pada Qos 1, sehingga

memungkinkan untuk diterapkan pada pengiriman data *real-time*, termasuk yang dilakukan pada penelitian ini.

Pada penelitian sebelumnya pengamanan data sinyal EKG dilakukan oleh Bramasta Agnanda Setiawan (2018) dengan metode anonimasi dan menggunakan algoritma *Jusak-Seedahmed* untuk mengembalikan struktur sinyal EKG setelah proses anonimasi. Proses anonimasi dilakukan dengan dengan menguraikan data dalam domain frekuensi. Anonimasi ini dilakukan dengan tujuan mengamankan data sinyal EKG dalam jaringan internet. Selanjutnya, penelitian ini dilanjutkan oleh Sony Solehuddin mengenai rekontruksi sinyal hasil dari proses anonimasi tersebut. Proses ini diawali dengan mengambil kunci terenkripsi yang disimpan pada *cloud server*. Pembuatan kunci ini diperoleh dari frekuensi rendah pada proses anonimasi. Proses rekonstruksi dilakukan berdasarkan algoritma FFT *(Fast Fourier Transform)* dengan melakukan transformasi bagian sinyal EKG dari *cloud server* dari domain waktu ke dmain frekuensi [3].

Pada tahun 2018, Ayaskanta Mishra[4] juga melakukan penelitian sistem pemantauan EKG jarak jauh berbasis MQTT menggunakan *Raspberry pi*, Modul AD8232 dan ADS1115. *Broker* MQTT yang digunakan adalah *CloudMQTT* yang merupakan *broker* online.

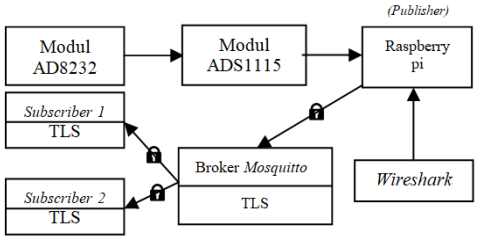
Pada tahun 2019, Hyunwoo Lee melakukan penelitian, dengan judul *"MQTLS: Toward Secure MQTT Communication with an Untrusted Broker"*. MQTLS ini merupakan protokol berbasis OpenSSL yang ditujukan untuk membatasi *broker* supaya tidak membaca pesan apa pun dari *publisher*, kecuali topik untuk pengiriman pesan[5]. Pada tahun yang sama penelitian mengenai MQTTS juga dilakukan oleh Neven Nikolov[6] dengan menggunakan sensor DHT22.

Pada tahun 2020, Adrian Febiyanto[7] melakukan penelitian mengenai pengamatan sinyal jantung EKG, dimana pada penelitian ini sinyal EKG akan dipantau dengan mengirimkan data pada *firebase* yang diakses secara *real-time*. Sistem pengiriman menggunakan *node sensor* berupa *raspberry pi* kemudian dari *firebase* akan dikirim ke aplikasi berbasis *mobile.*

Pada penelitian ini, protokol MQTT akan diterapkan pada pengiriman data EKG (Elektrokardiogram) yang merupakan data rekam aktifitas kelistrikan jantung. Penerapan protokol ini dengan mengaktifkan fitur keamanan pada MQTT yaitu dengan menggunakan protokol TLS. Protokol MQTT dan TLS biasa disebut dengan MQTTS. Penggunaan fitur keamanan ini dikarenakan data sinyal EKG merupakan data yang bersifat unik dan berbeda setiap individu.

Pada penelitian ini dilakukan analisis terhadap 3 parameter yaitu : selisih besar paket data sebelum enkripsi dan sesudah enkripsi, selisih waktu yang dibutuhkan sebelum pengiriman dan sesudah pengiriman, dan pengujian integritas data.

METODE PENELITIAN



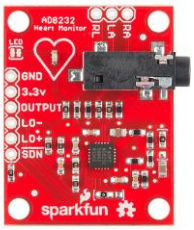
Gambar 1. Rancangan Penelitian

Berdasarkan gambar 1 di atas, terdapat modul AD8232, modul ADS1115 dan raspberry pi yang merupakan *node sensor* dan berperan sebagai *publisher*. Data yang dikirim berupa sinyal EKG yang sudah terenkripsi, dikarenakan pada saat pengiriman sudah menyertakan file kunci ca.crt yang sudah terpasang pada klien.

1. Perangkat *Node Sensor*

Node sensor pada penelitian ini terdiri dari beberapa komponen sebagai berikut :

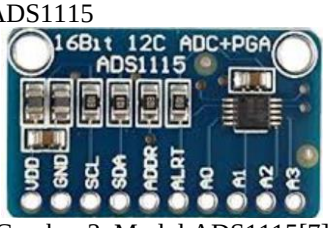
a. Modul AD8232



Gambar 2. Modul AD8232[7]

Modul AD8232 merupakan modul sensor Elektrokardiogram (EKG) yang bernilai analog. Untuk dapat mendeteksi sinyal jantung seseorang diperlukan elektroda penjapit.

b. Modul ADS1115



Gambar 3. Modul ADS1115[7]

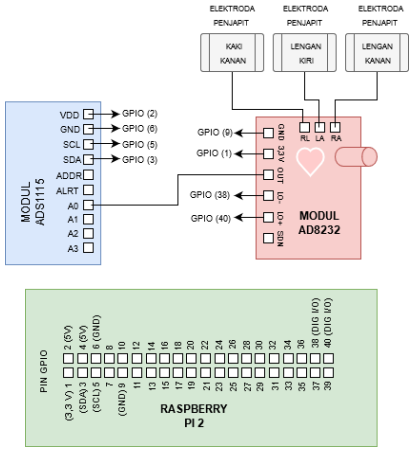
Modul ADS1115 merupakan modul adc *(analog digital converter)* 16 bit dengan tegangan sekitar 2,0-5,5 V. Modul ini juga memiliki fungsi PGA *(Programmable Gain Amplifier)* dimana fitur ini dapat memperkuat sinyal input analog pada saat pengambilan nilai ADC[7].

Modul ini digunakan untuk transfer data dari modul AD8232 ke *raspberry pi* supaya data sinyal EKG dapat terbaca pada *Raspberry pi.*

c. *Raspberry pi*

Seperti yang sudah banyak diketahui, *Raspberry pi* merupakan mini PC yang juga dapat

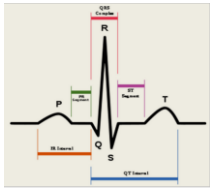
menggabungkan fungsi komputer dan mikrokontroler. Pada penelitian ini *Raspberry pi* akan terhubung oleh modul AD8232 dan ADS1115 sebagai node sensor. Untuk dapat menerima data yang ditransfer oleh ADS1115 diperlukan mengaktifkan fungsi I2C pada *Raspberry pi.* Berikut skematik rangkaian untuk *node sensor* :



Gambar 4. Skematik Rangkaian *Node Sensor*

Gambar 4 di atas merupakan sambungan *node sensor* untuk pengambilan data sinyal EKG. Masing – masing elektroda penjapit terhubung dengan 3 pin modul ad8232 yaitu RL*(Right Left)*, LA *(Left Arm)*, RA *(Right Arm)*. Kemudian pin OUT akan tersambung dengan pin A0 ADS1115 sehingga output akan ditransfer melalui I2C sehingga bisa terbaca oleh *Rapberry pi.*

2. Sinyal Elektrokardiogram (EKG)



Gambar 5. Sinyal EKG[7]

Sinyal Elektrokardiogram (EKG) merupakan sinyal yang merekam aktifitas kelistrikan otot jantung. Perekaman sinyal ini menggunakan elektroda penjapit di *lead* yang ditentukan. Pada penelitian ini menggunakan 3 titik untuk mengambil data sinyal jantung tersebut, dimana elektroda penjapit tersebut diletakkan pada lengan tangan kanan, lengan tangan kiri, dan kaki kanan. Berikut titik peletakan elektroda penjapit :

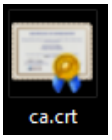
Gambar 6. Letak posisi elektroda penjapit

Seperti yang terlihat pada gambar 5, sinyal EKG terdiri dari 5 gelombang yaitu :

- Gelombang P
  adalah gelombang awal yang merupakan hasil depolarisasi di kedua atrium
- Gelombang Q
  merupakan gelombang defleksi negatif setelah gelombang P
- Gelombang R
  merupakan gelombang defleksi positif setelah gelombang P atau setelah gelombang Q.
- Gelombang S
  merupakan gelombang defleksi negatif setelah gelombang R atau gelombang Q
- Gelombang T
  merupakan hasil depolarisasi di kedua ventrikel. Normalnya positif dan *inverted* (terbalik) di aVR[8].

3. File Kunci (ca.crt)

Berdasarkan rangkaian penelitian pada gambar 1, data sinyal EKG yang telah dibaca oleh *node sensor Raspberry pi* akan langsung dikirim secara *real-time* ke kedua *subscriber* melalui *broker mosquitto* yang sudah terdapat file kunci (ca.crt).



Gambar 7. File kunci ca.crt

Pada sisi *publisher* juga menyertakan file kunci (ca.crt) untuk mengirimkan data. Data yang dikirim langsung terenkripsi. Proses enkripsi data berpengaruh pada besar paket data. Lama waktu proses enkripsi juga akan dianalisis melalui hasil rekam pada *wireshark* dengan menghitung selisih waktu saat pengiriman dan waktu sebelum tertangkap oleh *wireshark.* Selanjutnya, *broker* hanya bisa meneruskan data dari *client* yang memiliki kunci yang sama sehingga setiap pengirim dan penerima harus menyertakan file kunci (ca.crt) tersebut.

Pada sisi *subscriber* akan menerima data tersebut melalui terminal yang masing – masing sudah menggunakan fungsi *dump* untuk mengetahui besar paket data setelah didekripsi. *Subscriber* hanya menerima data yang sesuai dengan topik dan memiliki kunci yang sama. Pengiriman data akan dilakukan pada QoS 0 dan QoS 1. Dimana QoS 0 merupakan kualitas pelayanan yang hanya mengirimkan pesan tanpa memberi kepastian akan data sudah benar-benar tersampaikan atau tidak. Sedangkan pada QoS 1 data sudah terkirim dan tersampaikan dengan memberi *feedback* dari *subscriber* berupa SUBACK.

4. Pembuatan Kunci

Proses pembuatan file kunci (ca.crt) sendiri melalui openssl, dan dilakukan oleh broker. Kunci yang dibuat menggunakan algoritma enkripsi 3des, dengan panjang 2048 byte. Langkah – langkah pembuatan file kunci sebagai berikut:

1. *Open directory* openssl melalui terminal seperti berikut :
2. Membuat CA *key pair* untuk digunakan membuat CA *certificate* dengan syntax :
   `"openssl genrsa -des3 -out ca.key 2048"`
3. Membuat file CA *certificate* (ca.crt) akan digunakan untuk *publish* dan *subscribe. Syntax :*
   `"openssl req -new -x509 -days 1826 -key ca.key -out ca.crt"`
4. Membuat *server key pair* (server.key) yang akan digunakan oleh *broker*, dengan *syntax* :
   `"openssl genrsa -out server.key 2048"`
5. Kemudian membuat *server certificate request* (server.csr), dengan *syntax* :
   `"openssl req -new -out server.csr -key server.key"`
6. Kemudian membuat *server certificate* (server.crt), dengan menggunakan file ca.crt untuk verifikasi dan menandatangani file server.crt. *Syntax :*
   `"openssl x509 -req -in server.csr -CA ca.crt -CAkey ca.key -CA createserial -out server.crt -days 360"`

Setelah 6 file kuci tersebut dibuat. Kemudian masukkan *cafile* (ca.crt), *certfile* (server.crt, dan *keyfile* (server.key) pada *mosquitto.conf* seperti tampilan pada Gambar 8 berikut ini :

```
cafile /etc/mosquitto/certs/ca.crt
certfile /etc/mosquitto/certs/raspberrypidian.crt
keyfile /etc/mosquitto/certs/raspberrypidian.key
```

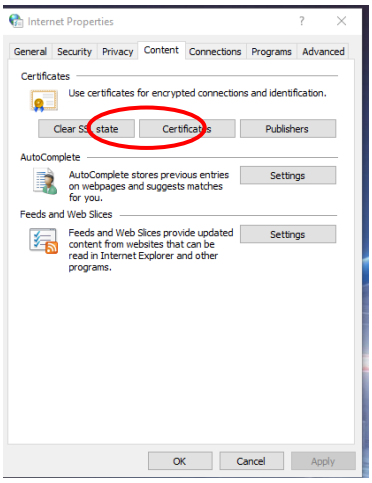
Gambar 8. File kunci pada *mosquitto.conf*

Nama *certfile* dan *keyfile* di sini ialah raspberrypidian.crt dan raspberrypidian.key.

5. Pemasangan File Kunci pada Klien

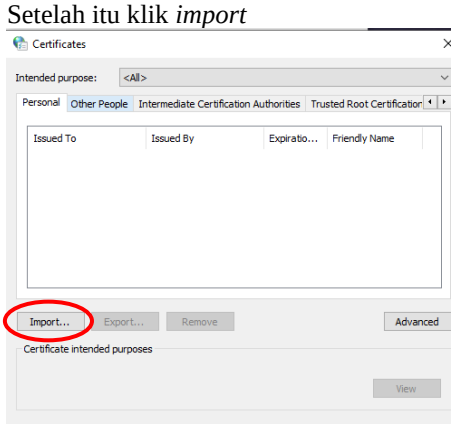
Setelah sudah dikonfigurasi pada mosquitto barulah membagikan file kunci ca.crt pada klien. Pemasangan file kunci pada klien menggunakan langkah –langkah berikut ini :

1. *Broker* membagikan file kunci ca.crt pada klien berupa PC
2. Cari *Internet Options*, akan muncul kotak dialog seperti dibawah ini :



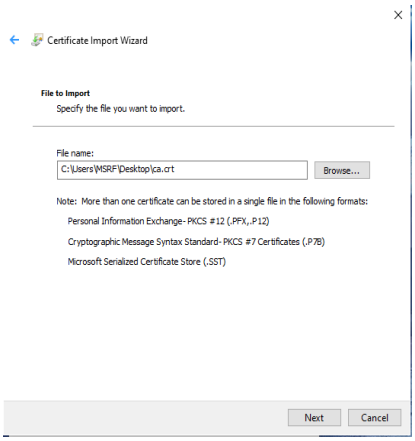
Gambar 9. Kotak dialog *Internet Options*

3. Kemudian klik *certificates*
4. Setelah itu klik *import*



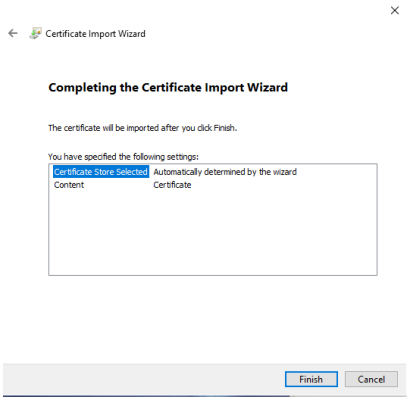
Gambar 10. *Insert certificates*

5. Kemudian arahkan *directory* file ca.crt



Gambar 11. *Import Certificates*

6. Kemudian klik *Next* → *Install*
7. Jika sudah selesai, klik *finish*



Gambar 12. *Certificate* telah di-*import*

Analisis data yang dilakukan pada penelitian ini meliputi 3 parameter yaitu :

1. Selisih besar paket data sebelum dan setelah pengiriman
2. Selisih waktu sebelum dan setelah pengiriman
3. Pengujian Integritas data menggunakan metode *cross-correlation* pada MATLAB

Perhitungan besar paket data bertujuan untuk mengetahui besar memori yang dibutuhkan untuk menggunakan MQTTS. Selisih waktu sebelum dan setelah pengiriman dilihat dari hasil rekam *wireshark* bertujuan untuk mengetahui rata-rata waktu yang dibutuhkan untuk proses enkripsi data pada MQTTS. Kemudian parameter ketiga yaitu pengujian integritas data dengan membandingkan data sebelum dikirim dan data setelah diterima menggunakan *cross-correlation.* Jika hasil lag ke-0 = 1 maka hasilnya sama, tidak berbeda meskipun data melewati proses enkripsi.

HASIL DAN PEMBAHASAN

Setelah melakukan pembuatan kunci pada *broker* dan pemasangan kunci pada klien. Percobaan pengiriman dapat dilakukan melalui terminal pada *publisher* dan *subscriber* dengan *syntax* sebagai berikut :

- *Syntax Publisher*

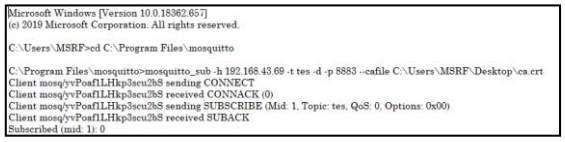
```
mosquitto_pub -h 192.168.43.121 -t tes
-d -p 8883 --cafile
C:\Users\Asus\Desktop\ca.crt
```

- *Syntax Subscriber*

```
mosquitto_sub -h 192.168.43.121 -t tes
-d -p 8883 --cafile
C:\Users\Asus\Desktop\ca.crt
```

Keterangan :

a. pub/sub merupakan perintah *publish* atau *subscribe*
b. –h merupakan *host* atau alamat *broker* yang dituju
c. –t merupakan topik yang digunakan
d. –d merupakan *dump* untuk mengetahui besar paket yang diterima, QoS yang digunakan, dan proses koneksi *broker*-klien
e. –p merupakan *port* yang digunakan. MQTTS terletak pada port 8883

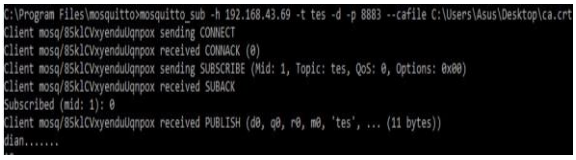


Gambar 13. *Subscriber* sudah terhubung dengan *Broker*

f. - -cafile merupakan letak file kunci ca.crt yang digunakan

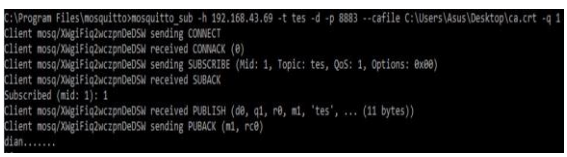
1. Percobaan Pengiriman pada Setiap QoS

Tes pengiriman pesan pada setiap QoS, dilakukan dengan menambahkan syntax "`-q`" dan hasilnya seperti Gambar 14 berikut ini :



Gambar 14. Pengiriman pada QoS 0

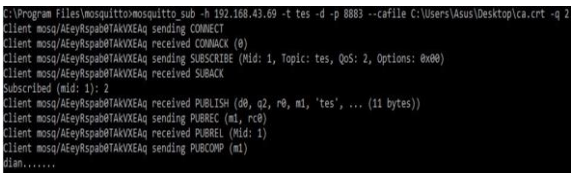
Perintah *subscribe* tanpa menuliskan *syntax* untuk QoS, maka akan otomatis menggunakan QoS 0 seperti gambar 14 diatas. Dari Gambar 14 diatas sinyal kontrol yang terdapat pada QoS 0 adalah PUBLISH kemudian dilanjutkan pesan yang diterima. Sinyal kontrol PUBLISH adalah pesan yang mewakili publikasi baru/terpisah.



Gambar 15. Pengiriman pada QoS 1

Pengiriman pada QoS 1 terdapat 2 sinyal kontrol yaitu : PUBLISH dan PUBACK, kemudian diikuti oleh pesan yang diterima.

PUBLISH sendiri sama dengan QoS 0, sedangkan PUBACK merupakan respons QoS 1 terhadap pesan publikasi. Pada setiap pengiriman baik QoS 0,1, atau 2 akan terlihat besar paket yang diterima, seperti yang terlihat pada gambar 14 dan 15 besar paket yang diterima adalah sebesar 11 byte. Data ini nantinya akan digunakan untuk analisis perbandingan paket data sebelum dan setelah terenkripsi.

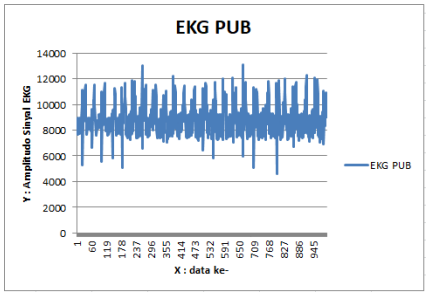


Gambar 16. Pengiriman pada QoS 2

Pada QoS 2 terdapat 4 sinyal kontrol yang digunakan yaitu : PUBLISH, PUBREC, PUBREL, dan PUBCOMP. PUBLISH terdapat di semua level QoS. Sinyal kontrol PUBREC merupakan bagian pertama dari aliran pesan QoS 2, PUBREL merupakan bagian kedua, dan PUBCOMP merupakan bagian terakhir dari aliran QoS 2, kemudian diikuti oleh pesan yang diterima. Apabila dari sisi *subscriber* tidak menerima sampai bagian terakhir dari QoS 2 (PUBCOMP) maka akan dikirimkan sinyal lagi supaya PUBCOMP tersampaikan pada *subscriber*. Alur protokol pada protokol QoS 2 memerlukan waktu yang sedikit lama dibanding dengan QoS 0 dan QoS 1.

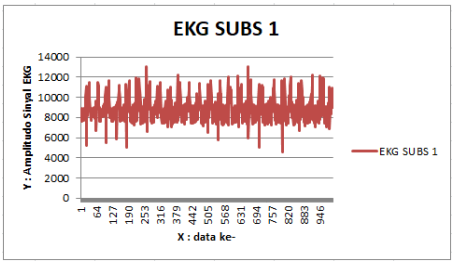
Pada penelitian ini tidak dilakukan pengiriman menggunakan QoS 2 dikarenakan waktu setiap pengirimannya yang lama sehingga menyebabkan beberapa data hilang *(lost)*. Tetapi disamping itu, kualitas pelayanan pada QoS 2 sangat terjamin bahwa data benar-benar tersampaikan pada sisi penerima dan pesan diterima 1 kali (tidak berulang).

2. Hasil Pengiriman pada QoS 0

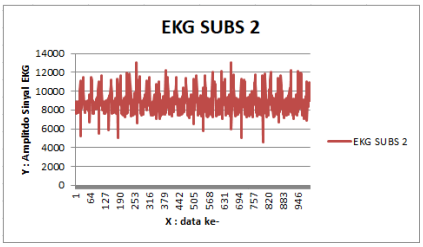


Gambar 17. Sinyal EKG sampel ke-1 yang dikirim

Pengiriman menggunakan sample sebanyak 10 orang dengan masing-masing 1000 data yang dikirim secara *real-time*, dan kedua *subscriber* menerima secara bersamaan.



Gambar 18. Sinyal EKG sampel ke-1 pada *Subscriber 1*



Gambar 19. Sinyal EKG sampel-1 pada *Subcriber 2*

Dari hasil pengirim dan penerimaan data pada kedua *subscriber*, dapat dilihat bahwa tidak ada perbedaan data yang signifikan berdasarkan grafik sinyal diatas.

a. Analisis Data

- Selisih besar paket kedua *subscriber*
  - ***Subscriber 1***

Tabel 1 Besar Paket Sebelum Enkripsi QoS 0

No Sampel	Besar Paket **Sebelum Enkripsi**		
	Min (byte)	Max (byte)	Rata- rata setiap sampel (byte)
1	4	5	4,142
2	4	5	4,135
3	4	5	4,088
4	4	5	4,12
5	4	5	4,161
6	4	5	4,119
7	4	5	4,004
8	4	5	4,298
9	4	5	4,009
10	4	5	4,164
Min/Max	4	5	
Standar Deviasi			0,0834
Rata-rata			4,124

Tabel 2 Besar Paket Setelah Enkripsi QoS 0

No Sampel	Besar Paket **Setelah Enkripsi**		
	Min (byte)	Max (byte)	Rata- rata setiap sampel (byte)
1	87	623	158,1918
2	87	584	149,9449
3	78	616	154,8506
4	87	619	160,8387

5	78	619	161,6364
6	87	615	155,9046
7	87	615	160,7799
8	87	622	163,1738
9	87	582	152,8293
10	87	616	149,5476
Min/Max	78	623	
Standar Deviasi			4,9222
Rata-rata			156,7698

Pada *subscriber 1* minimum besar paket sebelum enkripsi adalah 4 byte dan maksimal 5 byte. Sedangkan minimum besar paket setelah enkripsi sebesar 78 byte dan maksimum 623 byte. Rata – rata besar paket sebelum enkripsi sebesar 4,124 byte dengan standar deviasi sebesar 0,0834. Sedangkan rata-rata setelah enkripsi sebesar 156,7698 byte dengan standar deviasi sebesar 4,9222. Jika dihitung selisih rata-rata sebelum dan setelah enkripsi adalah 152,6458 byte.

- ***Subscriber 2***

Tabel 3 Besar Paket Sebelum Enkripsi QoS 0

No Sampel	Besar Paket **Sebelum Enkripsi**		
	Min (byte)	Max (byte)	Rata- rata setiap sampel (byte)
1	4	5	4,142
2	4	5	4,135
3	4	5	4,088
4	4	5	4,12
5	4	5	4,161
6	4	5	4,119
7	4	5	4,004
8	4	5	4,298
9	4	5	4,009
10	4	5	4,164
Min/Max	4	5	
Standar Deviasi			0,0834
Rata-rata			4,124

Tabel 4 Besar Paket Setelah Enkripsi QoS 0

No Sampel	Besar Paket **Setelah Enkripsi**		
	Min (byte)	Max (byte)	Rata- rata setiap sampel (byte)
1	87	483	146,4832
2	87	620	140,7121
3	87	285	141,2797
4	87	652	147,5339
5	87	615	147,4113
6	87	484	142,7909
7	87	615	145,1713
8	87	622	144,4837
9	87	516	139,7377
10	87	621	140,1403
Min/Max	87	652	
Standar Deviasi			3,0336
Rata-rata			143,5744

Hasil besar paket sebelum enkripsi *subscriber 2* sama dengan *subscriber 1*. Sedangkan hasil minimum besar paket setelah enkripsi adalah 87 byte, dan maksimum 652 byte. Rata-rata besar paket setelah enkripsi 143,5744 byte, dengan standar deviasi sebesar 3,0336. Selisih rata-rata sebelum dan setelah enkripsi sebesar 139,4504 byte.

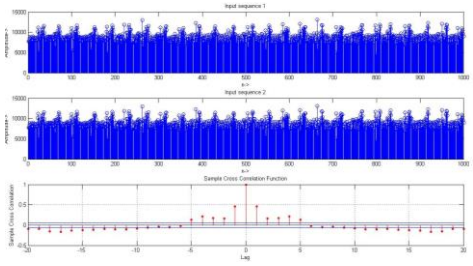
- Waktu yang dibutuhkan untuk proses enkripsi

Tabel 5 Waktu Enkripsi QoS 0

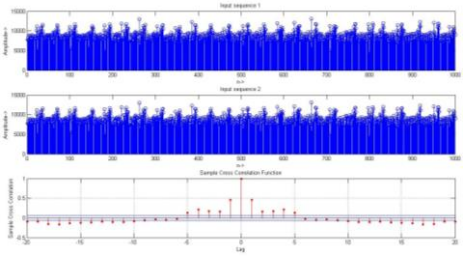
	No. Sampel	Rata-rata setiap sampel (s)	Waktu Min. (s)	Waktu Maks. (s)
Subscriber 1	1	0,001029	0,000111	0,213207
	2	0,000668	0,000096	0,120007
	3	0,000368	0,000132	0,015458
	4	0,000888	0,000111	0,135876
	5	0,001770	0,000105	0,215640
	6	0,001284	0,000099	0,268409
	7	0,001217	0,000105	0,232037
	8	0,000663	0,000106	0,040023
	9	0,000304	0,000125	0,007440
	10	0,000339	0,000107	0,022267
Subscriber 2	1	0,000315	0,000108	0,026047
	2	0,000647	0,000131	0,090009
	3	0,000557	0,000108	0,114175
	4	0,000897	0,000088	0,223816
	5	0,000758	0,000101	0,099999
	6	0,000266	0,000125	0,001883
	7	0,000469	0,000109	0,039982
	8	0,000925	0,000110	0,231028
	9	0,000730	0,000126	0,182455
	10	0,000296	0,000106	0,011076
Min/Max			0,000089	0,268409
Rata-rata keseluruhan	0,0007195 (s)			
Standar Deviasi	0,000395			

Pengiriman pada QoS 0 menghasilkan rata-rata keseluruhan dari kedua penerima yaitu 0,0007195 s. Dengan waktu minimum 0,000089 s dan maksimum 0,268409 s. Standar deviasi sebesar 0,000395.

b. Hasil *cross-correlation*



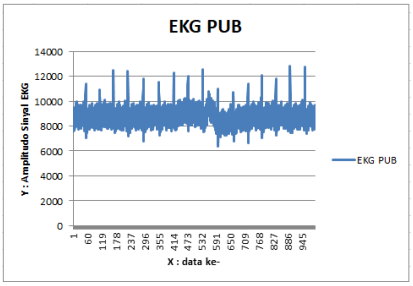
Gambar 20. Hasil *Cross-correlation subscriber* 1



Gambar 21. Hasil *Cross-correlation subscriber 2*

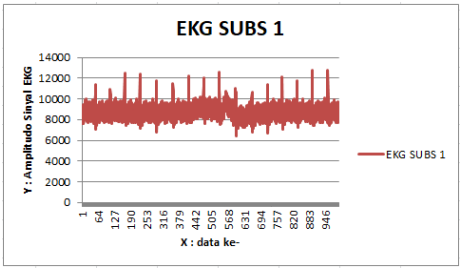
Hasil uji *cross-correlation* pada kedua *subscriber* menunjukkan hasil 1 pada lag ke-0, sehingga dapat diartikan data yang dikirim dan diterima adalah sama meskipun setelah melalui proses enkripsi.

3. Hasil Pengiriman pada QoS 1

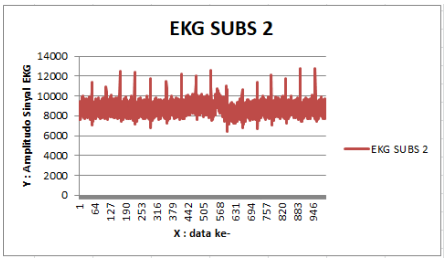


Gambar 22. Sinyal EKG sampel ke-10 yang dikirim

Tampilan Gambar 22 diatas merupakan salah satu dari 10 sampel data yang diambil. Sama halnya, dengan QoS 0 data antara pengirim dan penerima jika dilihat tidak ada perbedaan (sama).



Gambar 23. Sinyal EKG sampel ke-10 pada penerima 1 (QoS 1)



Gambar 24. Sinyal EKG sampel ke-10 pada penerima 2 (QoS 1)

a. Analisis Data

- Selisih besar paket kedua *subscriber*
- ***Subscriber 1***

Tabel 6. Besar Paket Sebelum Enkripsi QoS 1

No Sampel	Besar Paket **Sebelum Enkripsi**		
	Min (byte)	Max (byte)	Rata- rata setiap sampel (byte)
1	1	5	4,271
2	4	5	4,403
3	4	5	4,406
4	4	5	4,401
5	4	5	4,401
6	4	5	4,417
7	4	5	4,403
8	4	5	4,421
9	3	5	4,367
10	4	5	4,061
Min/Max	1	5	
Standar Deviasi			0,1122
Rata-rata			4,3551

Tabel 7 Besar Paket Setelah Enkripsi QoS 1

No Sampel	Besar Paket **Setelah Enkripsi**		
	Min (byte)	Max (byte)	Rata- rata setiap sampel (byte)
1	87	587	96,1483
2	89	656	96,2053
3	89	550	96,8126
4	89	551	95,1161
5	89	655	97,5437
6	89	726	104,9597
7	89	549	99,6813
8	89	727	101,8016
9	88	725	126,4734
10	89	720	126,7407
Min/Max	87	727	
Standar Deviasi			12,2068
Rata-rata			104,1483

Pengiriman QoS 1 pada *subscriber 1* menghasilkan rata-rata sebelum enkripsi adalah 4,3551 byte. Besar paket sebelum enkripsi minimum adalah 1 byte dan maksimum 5 byte, dengan standar deviasi sebesar 0,1122. Sedangkan besar paket setelah enkripsi menghasilkan minimum 87 byte dan maskimum 727 byte, dengan rata-rata 104,1483 byte. Standar deviasi yang dihasilkan sebesar 12,2068. Jika dihitung, selisih

rata-rata antara sebelum dan setelah enkripsi sebesar 99,7932 byte.

- ***Subscriber 2***

Tabel 8 Besar Paket Sebelum Enkripsi QoS 1

No Sampel	Besar Paket **Sebelum Enkripsi**		
	Min (byte)	Max (byte)	Rata- rata setiap sampel (byte)
1	1	5	4,271
2	4	5	4,403
3	4	5	4,406
4	4	5	4,401
5	4	5	4,401
6	4	5	4,417
7	4	5	4,403
8	4	5	4,421
9	3	5	4,367
10	4	5	4,061
Min/Max	1	5	
Standar Deviasi			0,1122
Rata-rata			4,3551

Tabel 9 Besar Paket Sebelum Enkripsi QoS 1

No Sampel	Besar Paket **Setelah Enkripsi**		
	Min (byte)	Max (byte)	Rata- rata setiap sampel (byte)
1	86	546	96,2407
2	89	727	98,1434
3	89	656	103,9379
4	89	726	118,1540
5	89	726	129,3213
6	89	728	141,8834
7	78	727	140,5599
8	89	657	131,8484
9	78	727	116,5707
10	89	650	122,2121
Min/ Max	78	728	
Standar Deviasi			16,4958
Rata-rata			119,8872

Pada *subscriber 2* besar paket sebelum enkripsi sama dengan *subscriber 1*. Sedangkan besar paket setelah enkripsi minimum sebesar 78 byte dan maksimum 728 byte, dengan standar deviasi 16,4958. Rata-rata besar paket setelah enkripsi sebesar 119,8872 byte. Selisih rata-rata antara sebelum dan setelah enkripsi sebesar 115,5321 byte.

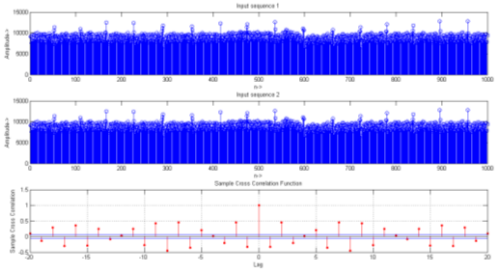
- Waktu yang diperlukan untuk proses enkripsi

Tabel 10 Waktu Enkripsi QoS 1

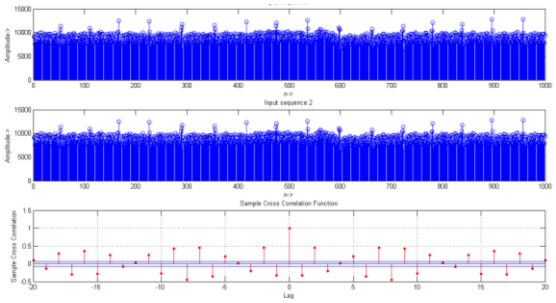
No Sam	Rata-rata setiap	Waktu Min. (s)	Waktu Maks. (s)
	Pel sampel (s)		
Subscriber 1	1 0,004594	0,000026	0,212009
	2 0,001119	0,000042	0,130032
	3 0,009943	0,000073	0,419589
	4 0,002654	0,000100	0,220023
	5 0,001929	0,000038	0,149941
	6 0,002021	0,000053	0,078267
	7 0,002210	0,000112	0,738426
	8 0,002098	0,000028	0,154626
	9 0,004467	0,000072	0,219614
	10 0,017735	0,000025	0,907781
Subscriber 2	1 0,001071	0,000035	0,218703
	2 0,008175	0,000021	0,346234
	3 0,000968	0,000063	0,101576
	4 0,012749	0,000050	0,508535
	5 0,018501	0,000067	0,543705
	6 0,027637	0,000107	0,682976
	7 0,017472	0,000033	0,189993
	8 0,036240	0,000088	0,911691
	9 0,013530	0,000079	0,966936
	10 0,006562	0,000042	0,229366
Min/ Max		0,000021	0,966936
Rata-rata keselu-Ruhan	0,009584 (s)		
Stan-dar Deviasi	0,009801		

Untuk rata – rata waktu proses enkripsi pada QoS 1 menghasilkan 0,009584 s. Dengan waktu minimum pengiriman antara kedua *subscriber* adalah 0,000021 s dan waktu maksimum adalah 0,966936 s, dengan standar deviasi sebesar 0,009801. Seperti yang dibahas pada sub bab sebelumnya dikarenakan pada QoS 1 terdapat tambahan sinyal kontrol yaitu PUBACK untuk memastikan bahwa data benar – benar sampai pada sisi penerima.

b. Hasil *Cross-correlation*



Gambar 25. Hasil *cross-correlation subscriber 1* (QoS 1)



Gambar 26. Hasil *cross-correlation subscriber 2* (QoS 1)

Hasil kedua *subscriber* menunjukkan pengiriman pada QoS 1 menghasilkan nilai 1 pada lag ke-0 , hal ini dapat diartikan data yang dikirim dan diterima adalah sama atau tidak ada perbedaaan.

KESIMPULAN

Berdasarkan pengujian pengiriman yang telah dilakukan pada penelitian ini dapat disimpulkan sebagai berikut :

1. Penerapan fitur keamanan protokol MQTT dapat dilakukan dengan menggunakan protokol TLS (MQTTS.)
2. Penerapan MQTTS pada *mosquitto* (*broker* lokal) menggunakan metode enkripsi simetris, sehingga pemakaian kunci untuk semua klien sama dengan *broker*.
3. Penerapan MQTTS untuk pengiriman data pada bidang kesehatan *(telehealthcare)* dapat dilakukan, sebagaimana pada penelitian ini yang menggunakan sampel data EKG.
4. Hasil pengujian selisih besar paket sebelum dan setelah enkripsi menunjukkan selisih yang cukup jauh lebih besar. Tetapi, dengan demikian pengiriman data menjadi lebih aman.
5. Pada pengujian waktu yang dibutuhkan untuk proses enkripsi pada QoS 0 dan QoS 1 menunjukkan hasil 0,0007195 detik dan 0,009584 detik. Waktu pengiriman QoS 1 lebih lama dikarenakan adanya tambahan sinyal kontrol yaitu PUBACK pada QoS 1.
6. Pengujian integritas data menggunakan metode *cross-correlation* pada QoS 0 dan QoS 1 menunjukkan hasil 1 pada lag ke-0. Sehingga dapat diartikan pengririman data menggunakan MQTTS QoS 0 maupun QoS 1 tidak terjadi perubahan (sama).

Adapun saran untuk pengembangan penelitian ini adalah menerapkan protokol MQTTS pada *broker online*, sehingga proses transmisi dapat dilakukan dalam jarak yang lebih jauh. Selain itu, dapat dilakukan untuk pengujian keamanan protokol MQTTS terhadap serangan – serangan dari luar.

DAFTAR PUSTAKA

[1] M. B. Yassein, M. Q. Shatnawi, S. Aljwarneh and R. Al-Hatmi, "Internet of Things: Survey and open issues of MQTT protocol," 2017 *International Conference on Engineering & MIS (ICEMIS)*, Monastir, 2017, pp. 1-6, doi: 10.1109/ICEMIS.2017.8273112.

[2] M. R. Bintami, Rancang Bangun Transmisi Data Heart Rate menggunakan Protokol MQTT, Surabaya: Universitas Dinamika, 2019.

[3] B. A. Setiawan, I. Puspasari, S. Solehuddin dan J. Jusak, "Implementasi Pengamanan Transmisi Sinyal EKG (Elektrokardiogram) secara Daring dengan Metode Anonimasi," *Elkomika*, vol. 7, pp. 85-96, 2019.

[4] A. Mishra, A. Kumari, P. Sajit dan P. Pandey, "Remote Web Based ECG Monitoring Using MQTT Protocol," *International Journal of Advance Engineering and Research*, vol. 5, no. 4, p. 1100, 2018.

[5] H. Lee, J. Lim dan T. Kwon, "MQTLS: Toward Secure MQTT Communication with an Untrusted *Broker*," *International Conference on Information and Communication Technology Convergence (ICTC)*, p. 53, 2019 .

[6] N. Nikolov and O. Nakov, "Research of Secure Communication of Esp32 IoT Embedded System to.NET Core Cloud Structure using MQTTS SSL/TLS," *2019 IEEE XXVIII International Scientific Conference Electronics (ET)*, Sozopol, Bulgaria, 2019, pp. 1-4, doi: 10.1109/ET.2019.8878636.

[7] A. Febiyanto, Pengukuran dan Pengamatan Sinyal Electrocardiogram menggunakan Raspberry dengan Tampilan Aplikasi Mobile, Surabaya: Universitas Dinamika, 2019.

[8] B. A. Setiawan, Anonimasi Sinyal EKG (Elektrokardiogram) untuk Keamanan Transmisi Data pada Sebuah *Node Sensor*, Surabaya: Universitas Dinamika, 2018.