Sistem Keamanan Data Pada IoT Berbasis MQTT Dan Database MySQL Menggunakan Metode RSA

1st Aldi Dwi Febriyanto
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
aldiidf@student.telkomuniversity.ac.id

2nd Sofia Naning Hertiana
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
sofiananing@telkomuniversity.ac.id

3rd Yudha Purwanto
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
omyudha@telkomuniversity.ac.id

Abstrak—Dibalik kemajuan teknologi yang pesat menyimpan ketakutan terhadap pencurian data. Adapun solusi untuk mengatasi pencurian data tersebut sistem seharusnya memakai Kriptografi. Pada penelitian tugas akhir ini akan dirancang suatu sistem yang dapat menjaga keamanan pada proses pengiriman data melalui protokol MQTT penyimpanan data pada PHP MyAdmin dengan Metode RSA. Berdasarkan hasil pengujian sistem, diketahui bahwa sistem dapat melakukan enkripsi pada data sensor dan dapat mengirimkan data dalam bentuk chipertext melalui MQTT menuju database. Nilai rata-rata response times untuk skenario RSA 1024 bit sebesar 879,4 ms untuk proses enkripsi dan 2189,83 ms untuk proses dekripsi. Untuk skenario RSA 2048 bit didapatkan hasil rata-rata pengujian Response Times untuk 10, 20, dan 30 data sebesar 2.859,27 ms untuk proses enkripsi dan 228.060,91 ms untuk proses dekripsi. Proses pengiriman data dari sensor menuju database menunjukan bahwa QoS pada sistem yang telah dibuat termasuk kategori baik dengan masing-masing nilai rata-rata untuk skenario RSA 1024 bit dan RSA 2048 bit yaitu throughput 6932,04 Kb/s dan 6167,03 Kb/s, packet loss sebesar 0% serta delay sebesar 243,3 ms dan 589,5 ms.

Kata kunci— RSA, internet of things, sistem keamanan, enkripsi, dekeripsi, MQTT.

I. PENDAHULUAN

Seiring berjalan waktu *Internet of Things* (IoT) banyak digunakan dan dikembangkan dengan pesat, dengan semakin mudah nya seseorang untuk memperoleh informasi dari berbagai sumber. Proses transfer data dan proses penyimpanan data di dalam *database* juga semakin berkembang. Semakin maju teknologi dalam mengakses informasi kerap disalahgunakan oleh berbagai pihak yang tidak bertanggung jawab [1]. Informasi tersebut lah yang harus diberikan keamanan karena tidak menutup kemungkinan informasi tersebut dapat diakses oleh pihak yang tidak bertanggung jawab dan menyebabkan kerugian bagi instansi tertentu. Oleh sebab itu penulis merancang sistem keamanan agar informasi tersebut tidak disalahgunakan.

Dengan latar belakang ketakutan tindakan pencurian data pada saat pengiriman dan didalam *database*,penulis merancang sebuah sistem yang dapat menghindari manipulasi dokumen yang dapat disalah gunakan untuk kepentingan tertentu.

Terdapat beberapa cara pengamanan melalui proses autentifikasi dengan dukungan kriptografi yang diantaranya adalah penggunaan algoritma Hash dan pengunaan *Cipher* Simetrik dan Asimetrik. Algoritma hash sangat berguna jika user ingin melakukan pengamanan pesan secara *irreversible* (satu arah). Sedangakan metode *Cipher* simetrik dan asimetrik berguna untuk pengaman yang bersifat *reversible* (dua arah) [6].

Peneliti menggunakan algoritma RSA untuk melakukan pengamanan data sensor yang dikirim melalui MQTT dan *database* PHP My Admin. Data yang sudah di enkripsi akan dikirimkan melalui MQTT dan pada saat didalam *database* akan dilakukan dekripsi secara otomatis.

II. KAJIAN TEORI

A. Algoritma Kriptografi

Dalam ilmu kriptografi, terdapat dua buah proses yaitu melakukan enkripsi dan dekripsi. Pesan yang akan dienkripsi disebut sebagai *plaintex*t (teks biasa) [14]. Disebut demikian karena informasi ini dengan mudah dapat dibaca dan dipahami oleh siapa saja. Algoritma yang dipakai untuk mengenkripsi dan mendekripsi sebuah *plaintext* melibatkan penggunaan suatu bentuk kunci. Pesan *plaintext* yang telah dienkripsi dikenal sebagai *ciphertext*. Kriptografi terdiri dari berbagai komponen pendukung, seperti:

1. Enkripsi

Enkripsi sangat penting dalam kriptografi, yaitu metode untuk melindungi data yang dikirim agar tetap rahasia. Pesan aslinya disebut Teks biasa, diubah menjadi kode-kode yang sulit dipahami.

2. Dekripsi

Dekripsi merupakan kebalikan dari enkripsi. Pesan yang telah dienkripsi dikembalikan ke bentuk asalnya. Algoritma yang digunakan untuk dekripsi tentu berbeda dengan yang digunakan untuk enkripsi.

3. Kunci

Adalah kunci yang dipakai untuk melakukan enkripsi dan dekripsi. Kunci terbagi menjadi dua bagian, yaitu *private key* dan *public key*.

4. *Chipertext*.

Merupakan suatu pesan yang telah melalui proses enkripsi. Pesan yang ada pada teks kode ini tidak bisa

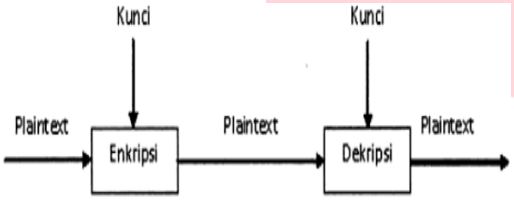
dibaca karena berupa karakter-karakter yang tidak mempunyai makna.

5. *Plaintext*.

Sering disebut dengan teks asli atau teks biasa ini merupakan pesan yang diketik yang memiliki makna. Teks asli inilah yang diproses menggunakan algoritma kriptografi untuk menjadi *chipertext* (teks-kode).

6. *Cyptanalysis*.

Ini dapat dipahami sebagai analisis kode atau sains, dan teks asli dapat diperoleh tanpa mengetahui kunci yang valid. Jika teks kode berhasil diubah ke teks asli tanpa menggunakan kunci yang valid, prosesnya disebut *cracking* kode. Ini dilakukan oleh seorang *cryptanalyst*.



GAMBAR 2. 1
SKEMA ENKRIPSI DAN DEKRIPSI DENGAN MENGGUNAKAN KUNCI.

B. Algoritma RSA

RSA merupakan algoritma kriptografi kunci publik (asimetris). Ditemukan pertama kali pada tahun 1977 oleh Ron Rivest, Adi Shamir, dan Len Adleman. Nama RSA sendiri diambil dari ketiga penemunya tersebut. Sebagai algoritma kunci publik, RSA mempunyai dua kunci, yaitu kunci publik dan kunci rahasia. RSA mendasarkan proses enkripsi dan dekripsinya pada konsep bilangan prima dan aritmetika modulo. Baik kunci enkripsi maupun dekripsi keduanya merupakan bilangan bulat. Kunci enkripsi tidak dirahasiakan dan diberikan kepada umum sehingga disebut dengan kunci publik, namun kunci untuk dekripsi bersifat rahasia (kunci *privat*).

Tingkat keamanan algoritma penyandian RSA sangat tergantung pada ukuran kunci sandi (dalam bit). Semakin besar ukuran kunci, maka semakin sulit juga penyadap terjadi. Kombinasi kunci ini lebih dikenal dengan istilah *brute force attack* yang bila panjang kuncinya 256 bit, maka menjadi tidak ekonomis dan sia-sia jika *hacker* pun meyadap sandi [15]. .

C. Pembuatan Kunci

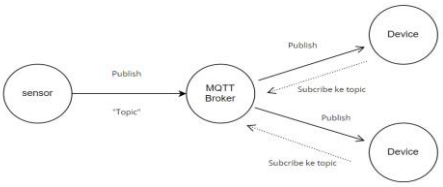
Algoritma pembangkit kunci ada 2 bilangan prima besar yaitu n = p x q sangat sulit untuk difaktorisasikan. Di rekomendasikan besar p dan q adalah 512 bit sehingga n berukuran 1024 bit. Karena p dan q adalah bilangan prima, Langkah-langkah pembuatan kunci RSA [14]:

1. Pilih dua buah bilangan prima sembarang p dan q. nilai p dan q harus dirahasiakan.
2. Hitung nilai n dari rumus, n – p x q. Besaran n tidak perlu dirahasiakan
3. Hitung m = (p – 1) (q – 1). Bersaran m perlu dirahasiakan
4. Dipilih sebuah bilangan bulat sebgai kunci publik, disebut namanya e, yaitu relatif prima terhadap m. e relatif prima terhadap m artinya faktor pembagi terbesar keduanya adalah 1, secara matematis disebut gcd (e,m)=1. Untuk mencarinya dapat digunakan algoritma Euclid. Nilai e bersifat tidak rahasia.

D. MQTT

Message Queue Telemetry Transport (MQTT) adalah sebuah protokol komunikasi data *machine to machine* (M2M) yang berada pada layer aplikasi, *MQTT* bersifat *lightweight message* artinya MQTT berkomunikasi dengan mengirimkan data pesan yang memiliki *header* berukuran kecil yaitu hanya sebesar 2 bytes untuk setiap jenis data, sehingga dapat bekerja di dalam lingkungan yang terbatas sumber dayanya seperti kecilnya *bandwidth* dan terbatasnya sumber daya listrik, selain itu protokol ini juga menjamin terkiriminya semua pesan walaupun koneksi terputus sementara, protokol *MQTT* menggunakan metode *publish*/*subscribe* untuk metode komunikasinya [5]. *Publish*/*subscribe* sendiri adalah sebuah pola pertukaran pesan di dalam komuunikasi jaringan dimana pengirim data disebut *publisher* dan penerima data disebut dengan *subscribe*.

Pengiriman data pada *MQTT* didasari oleh topik, topik ini nantinya yang akan menentukan pesan dari *publisher* harus dikirim pada *subscriber* yang mana, topik ini dapat bersifat hirarki, *MQTT* topik memiliki tipe data string dan untuk perbedaan hirarki atau level dari topik digunakan tanda baca “/” [11].



GAMBAR 2. 2
SISTEM KERJA PROTOKOL MQTT.

E. Database

Database merupakan kumpulan informasi atau data yang terintegrasi dengan baik yang tersimpan di dalam komputer. Untuk mengolah *database* diperlukan sebuah perangkat lunak yang disebut *Database Management System* (DBMS). DBMS merupakan suatu sistem perangkat lunak yang memungkinkan pengguna untuk membuat, memelihara, mengontrol dan mengakses *database* secara praktis dan efisien [9].

1. MySQL

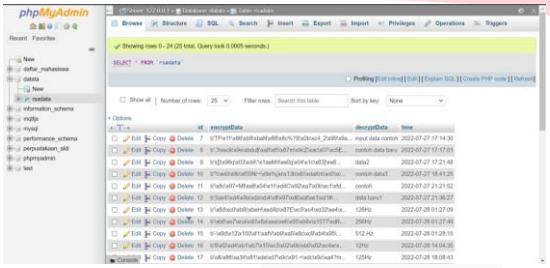
MySQL adalah salah satu jenis *database* server yang

sangat terkenal. Kepopulerannya disebabkan Mysql menggunakan SQL sebagai bahasa dasar untuk mengakses databasenya. Mysql termasuk jenis *Relational Database Management System* (RDBMS). Pada Mysql, sebuah *database* mengandung satu atau sejumlah tabel. Tabel terdiri atas sejumlah baris dan setiap baris mengandung satu atau beberapa kolom. Untuk mengelola *database* Mysql ada

beberapa cara yaitu melalui *prompt DOS* (*tool command line*) [8].

2. PHP MyAdmin

PHP *myAdmin* merupakan *front-end* mysql berbasis web dan dibuat dengan menggunakan PHP [9]. Php *myAdmin* mendukung berbagai fitur administrasi mysql termasuk manipulasi *database*, tabel, *index* dan juga dapat mengeksport data ke dalam berbagai format data. Tampilan PHP *myAdmin* seperti gambar 2.3.



GAMBAR 2. 3

TAMPILAN PHP *MYADMIN*.

F. Raspberry Pi

Raspberry Pi merupakan mini komputer atau *microcomputer* seukuran kartu kredit yang dapat digunakan untuk menjalankan program perkantoran, permainan komputer dan sebagai pemutar media. Raspberry Pi dikembangkan oleh yayasan nirlaba *Raspberry Pi Foundation* dengan tujuan untuk meningkatkan minat pendidikan sains komputer. Komputer berukuran kecil dengan nilai yang bisa dijangkau oleh masyarakat ini bisa menjadi alat yang multifungsional. Terdapat juga konektor *input* dan *output* serta perangkat keras komputer itu sendiri pada papan sirkuit cetak. Raspberry Pi model B ini juga memiliki *port* HDMI dan keluaran video komposit, empat buah *port* USB tipe 2.0, satu buah *port* Ethernet 10/100, dan *slot* kartu microSD, serta konektor bertipe GPIO serta pada audio disertai output audio bertipe *analog* [16].

G. Sensor Ultrasonik HC-SR04

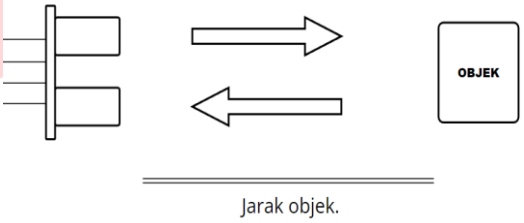
Sensor Ultrasonik HC-SR04 adalah perangkat yang sering digunakan untuk mengukur jarak dari suatu objek. Sensor ini memiliki 2 komponen yaitu ultrasonik *transmiter* sebagai pemancar suara dan ultrasonik *receiver* sebagai penerima suara. Jarak maksimal yang dapat diukur sekitas 3-4 meter [2]. Sensor ultrasonik HC-SR04 memiliki 4 pin, yaitu:

1. VCC, pin ini akan dihubungkan pada tegangan 5 volt.
2. Tring, pin ini berfungsi sebagai pengirim gelombang.
3. Echo, pin ini berfungsi sebagai penerima gelombang pantul.
4. GND, pin ini akan dihubungkan pada ground.

H. Prinsip Kerja Sensor Ultrasonik HC-SR04

Pada gambar 2.4 dapat dijelaskan bahwa sensor ultrasonik akan memancarkan gelombang yang memiliki frekuensi 40 kHz, lalu gelombang yang dipancarkan akan merambat sebagai gelombang bunyi dengan kecepatan

310 m/s. Ketika gelombang tersebut mengenai objek, maka gelombang tersebut akan dipantulkan oleh objek, lalu setelah gelombang pantul sampai ke alat penerima, maka gelombang tersebut akan diproses untuk mengetahui jarak dari suatu objek tersebut.

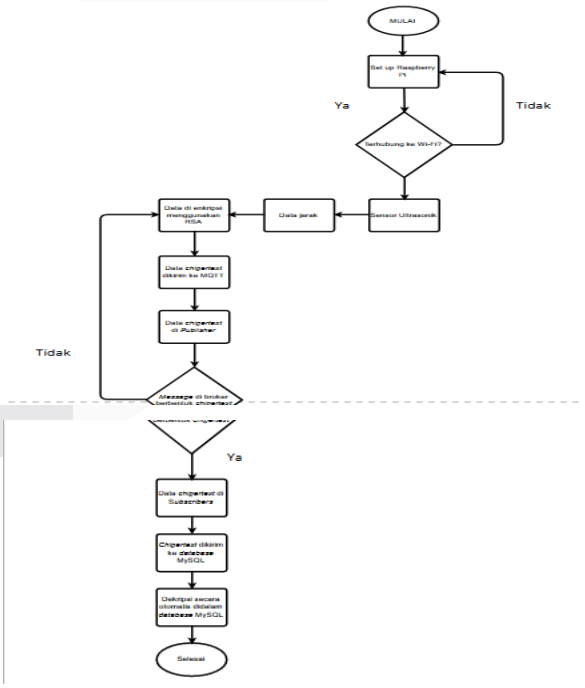


GAMBAR 2. 4

PRINSIP KERJA SENSOR ULTRASONIK HC-SR04.

III. METODE

A. Diagram Alir Sistem



GAMBAR 3. 1

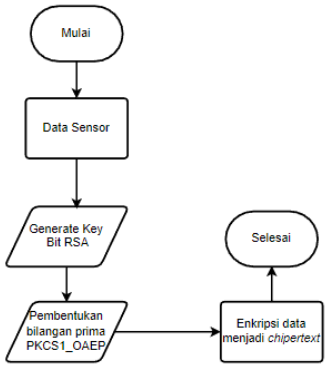
DIAGRAM ALIR SISTEM UMUM.

Dalam Gambar 3.1, menjelaskan bahwa bagaimana proses simulasi ini akan dilakukan. Awalnya, sensor ultrasonik

mendeteksi jarak, setelah itu data sensor ultrasonik di enkripsi menggunakan metode RSA menjadi *chipertext.* Data yang sudah berbentuk *chipertext* di masukkan ke dalam MQTT. Data yang telah di masukkan akan di simpan sementara dalam MQTT *broker*. Apabila data yang ditampilkan didalam *broker* masih berbentuk plaintext maka akan dilakukan enkripsi ulang, namun apabila data yang ditampilkan didalam MQTT *broker* sudah berbentuk *chipertext* maka akan dikirimkan oleh *subscribers* kedalam *database* MySQL. Didalam *database chipertext* akan di dekripsi secara otomatis sehingga data yang di tampilkan didalam *database* berupa *chipertext* dan dekripsinya.

B. Diagram Alir Algoritma RSA

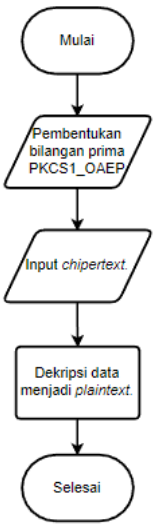
Diagram alir algoritma enkripsi RSA dapat dilihat pada gambar 3.2.



GAMBAR 3. 2

DIAGRAM ALIR ENKRIPSI RSA.

Gambar 3.2 menunjukan bagaimana proses enkripsi algoritma RSA yang diawali dengan input data sensor. Setelah itu menentukan panjang kunci RSA yang akan digunakan. Lalu dilakukan pembangkitan kunci publik menggunakan PKCS1_OAEP untuk melakukan enkripsi dengan menentukan bilangan prima yang akan dilakukan secara otomatis oleh sistem. Kemudian enkripsi diproses sampai akhirnya menghasilkan *chipertext*.

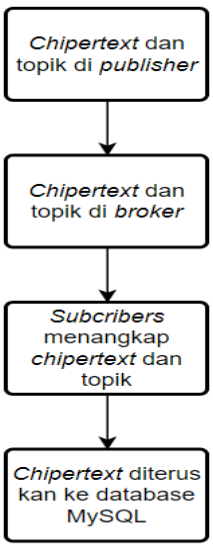


GAMBAR 3. 3

DIAGRAM ALIR DEKRIPSI RSA.

Gambar 3.3 merupakan proses dekripsi yang dimulai dengan pembentukan kunci *private* menggunakan PKCS1_OAEP dengan menentukan bilangan prima yang akan dilakukan secara otomatis oleh sistem. Kemudian input pesan yang sudah dienkripsi (*Chipertext).* Pendekripsian teks diproses hingga menghasilkan *plaintext* kembali.

C. Diagram Alir MQTT



GAMBAR 3. 4

DIAGRAM ALIR MQTT.

Pada gambar 3.4 dijelaskan diagram alir dari proses MQTT. *Publisher* MQTT mengirimkan data *chipertext* dan topik ke *broker*. Setelah itu *subcribers* akan menambahkan atau menangkap *chipertext* melalui topik yang sudah ditentukan. Apabila topik yang ditambahkan oleh *subcribers* tidak sesuai dengan topik yang ditentukan *publisher* maka data tidak akan sampai di *subcribers*. Setelah itu data yang masih berbentuk *chipertext* akan dikirimkan kedalam *database* MySQL.

D. Spesifikasi Perangkat Penelitian

1. Sarana Perangkat Keras

Sarana perangkat keras yang digunakan berupa laptop, mikro komputer dan sensor dengan spesifikasi sebagai berikut:

TABEL 3. 1

SPESIFIKASI PERANGKAT KERAS.

No.	Jenis	Tipe
1.	Unit Pemroses Sentral	AMD Ryzen 3 3200U with Radeon Vega Mobile Gfx 2.60 GHz.
2.	Memori Akses Acak	8,00 GB.
3.	Grafik	NVIDIA GeForce MX230.
4.	Mikro Komputer	Raspberry Pi 4 model B.
5.	Sensor Ultrasonik	HC-SR04.

2. Sarana Perangkat Lunak

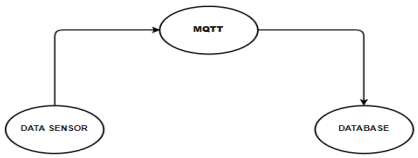
Perangkat lunak ini merupakan sarana yang berupa program komputer. Perangkat lunak yang digunakan untuk menunjang penelitian ini dapat dilihat pada Tabel 3.2.

TABEL 3. 2

SPESIFIKASI PERANGKAT LUNAK.

No.	Jenis	Tipe
1.	Sistem Operasi	Windows 10 Home Single Language.
2.	Aplikasi	Visual Studio Code dan Command Prompt.

E. Perancangan Topologi Jaringan



GAMBAR 3. 5

TOPOLOGI JARINGAN.

Berdasarkan gambar 3.5 terlihat jika alur dimulai dari pengiriman data sensor menuju protokol MQTT. Setelah itu data didalam MQTT akan diteruskan ke *database* melalui *broker* dalam bentuk *chipertext*. Didalam *database* data akan ditampilkan dalam bentuk *chipertext* dan akan dilakukan dekripsi secara otomatis menjadi *plaintext.*

F. *Quality of Service* (QoS)

Pengujian ini dilakukan menggunakan beberapa parameter, yaitu *delay* dan *packet loss*. Pengujian pengukuran ini dilakukan menggunakan aplikasi wireshark dan mengikuti standarisasi ITU-T G1010. Table standarisasi QoS yang dipakai adalah sebagai berikut.

TABEL 3. 3

SPESIFIKASI STANDAR MENURUT ITU-T G.1010 [17].

Medium	***Application***	**Degree of symmetry**	**Typical amount of data**	**Key performance parameters and target values**		
				One-way delay (Note)	**Delay variation**	**Information loss**
Data	Low priority transactions	Primarily one-way	< 10 KB	< 30 s	N.A.	Zero
Data	Command/ control	Two-way	~ 1 KB	<250 ms	N.A.	Zero

G. *Response Times Testing*

Pengujian *Response Times* dilakukan untuk mengetahui apakah sistem enkripsi dan dekripsi yang dilakukan sudah termasuk standarasi baik atau tidak.

TABEL 3. 4

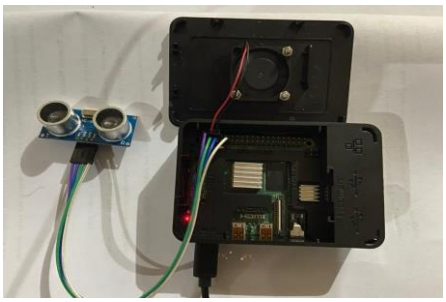
TABEL STANDAR PENGUJIAN *RESPONSE TIMES* [18].

Waktu *Response*	**Keterangan**
< 0,1 s	Baik
0,1 s – 6 s	Cukup
>6 s	Buruk

IV. HASIL DAN PEMBAHASAN

A. Hasil Rancangan Alat

Perangkat keras yang digunakan pada Tugas Akhir ini terdiri dari sensor ultrasonik HC-SR04, V-Gen 8GB Micro SD, dan mikrokomputer Raspberry Pi 4 model B. Sensor diarahkan ke sekitar untuk mendapatkan hasil data dari sensor.



GAMBAR 4. 1
PERANCANGAN ALAT.

B. Hasil Pengujian dan Analisis Sistem

1. Analisis Pengujian *Avalanche Effect*

Pada pengujian *Avalanche Effect* yang dilakukan pada sistem menggunakan RSA 1024 dan 2048 bit dengan mengganti 1 bit pada *plaintext.* Berikut variabel uji yang digunakan oleh algoritma tersebut menggunakan scenario yang telah ditentukan sebelumnya.

TABEL 4. 1
HASIL PENGUJIAN RSA 1024 BIT.

Testcase 1	RSA 1024 Bit.
Public Key	*Generate Random Primes*
Private Key	*Generate Random Primes*
Plaintext	60.201250140093491
Chipertext	*;\xf6\xda\xfe,R\xd2#\xce\xe5\x89\x05\xafG\xc3\xbb\x06\x88\t\xdf\x00\xd8X\x8d\x07\xc0>3*\xb2ri\xd7\xbb}\xb6\xd6\xaa\xa3\xf8Tt\xd9\\xf0(f\xac\xed/\x0bO\xd3\x9e7i^\xd7\xfe\x02\x80\x02|\xeb\xf5\x1c\x07\xe2\x06%\xa9];\xcfF\x9epR^\x9e\x04Mt\xab\xfb@\xd0\x8b\x07%\x16\x8fE\xe4\x02*\xc4m\x822\xeb\\\xeb\xf5\x15\xe0&;\x00\x96\x88\xbdpSLouV\x82\xbf\x0b\x1a\xc1(z\x0f\x8ca
Decrypt Text	60.201250140093491
Testcase 2	RSA 1024 Bit
Public Key	*Generate Random Primes*
Private Key	*Generate Random Primes*
Plaintext	65.201250140093491
Chipertext	F\\x8e\\x9c<\\x12\\xc7\\x07\\xe6\\x87\\x1fA\\xe8\\xde\\xf3\\x1ab\\x16Bb\\xff\\x03\\x95E\\xa0\\xb9\\x06\\xd4\\xc0\\xe2\\xc7\\x81\\xf2c\\xb5\\xe0\\x8a\\x80\\xb5\\xb6\\x895\\xe7\\x1cF\\xa9^p\\x94\\x18\\x90\\xb1\\xcf;M\\x982\\xa334\\x82\\x9a\\x99F\\x11O\\xeer\\xd8\\xdeY\\xd1\\xdd2\\xc1\\xc6D^PW\\xc22\\x9f\\xe5=\\xe6\\x9d\\xba?\\x17\\x19\\x84\\xa1~p\\x98L\\x803i\\xbf\\x11\\xb3\\xf9\\x86\\xc4\\n}\\x8b\\x1e\\x99\\xc5r\\xf8TcO\\x995\\x00\\xb7\\x9c\\x00\\xa7G\\xd3;\\xbc
Jumlah bit *chipertext* berbeda	30/128 x 100% = 23,43%
Decrypt Text	65.201250140093491

TABEL 4. 2
HASIL PENGUJIAN RSA 2048 BIT.

Testcase 1	RSA 2048 Bit.
Public Key	*Generate Random Primes*
Private Key	*Generate Random Primes*
Plaintext	60.201250140093491
Chipertext	[\n\xe1'"\xf7u\x04\xee>\xf3;\x1cj\xb6~\xf0\x00cm\xebP\xf5G\xd2\xdc/\x83\x8fw\x89u\x94\xad\xad5\x17e\xc6\xf1\x96\x0f\xf9i$\xf7G\xe0o\xa3\xf3\xd2\xe9?\xff\xd9\xbeO"\xb180\xd5\xaeWG^P\x91\x14\xeb\x9cQ\x08\x0e\xf3Tff\xfb\x16\x93\x188\xbd=\tlQ\xad\xd4\x85\x14\x80\x16\x07]\xaaH\xaf\xd6\xae\xaf\xf5\|~\x18\xc0\x12~j\xa9\xc1(\x13\xd2po\x83N"\x15\xc6f6\xc5\x81\x1bo\x8d\x7f\xe2A,e>\x98)\xaa\xc6$o\x08\xd6\xe6M8\xa5(\x18m\xac\xcft\xa9\xa7\xad\xbf\xdd\x9a{\x90\xfa\xe3]\n\x89\x04/\x926\xd7\xa8\xce\x9a\xe5\xfc\x95\xb7\xa0\xa7x\x87J\x1c\x1d\x10\x00,\x83!\xef\xf4T\x11\xe2\x0c\x7f\x8ce!\xba$w\x13XYg"\x00R{\xc4\xbc/\xfa>\xef\xee\x12\xe88\xe5\x8f+\xb7\x9e<\xd0\xc0\x03\xaa\xad\xc8\xe2\xd3\x01S\x1a\x06\xb1i\x9e\x83\xb3\x04\xe1&e\xb1\xbc\xa4\xe7Y J
Decrypt Text	60.201250140093491
Testcase 2	RSA 2048 Bit.
Public Key	*Generate Random Primes*
Private Key	*Generate Random Primes*
Plaintext	65.201250140093491
Chipertext	\x8b\xee\x14\xe7\xc8\xd7\xfc\xba\x1bI\x89\xfek\n^b\xee\xd7F\x0cN}Y\xd5\xd6\xab\x02\xde\xd7P\x10\xa8LO\x91\x8a}\xc4?\x8f\xb4/+s\xe5g\xd7)1;+\x92v4\xa7\x18\x97;,*\xdb\xeeJ\x1bZ\xebs\xd0v\xffx\x02\xb2\xf9\x15\x14\xc7\xd0\xc0\xdb\xb2\x88\xa1\x0e\xcd\x16\xeb\xc1\xdc\xc7EC\xc3:FEH\\\x11v\x04\xd7FV`\xad\xd24\xe0\x08R\x19\x8f\xd8C\x1f\xef\xe0F\x1dW\xbc\xcb\x19\xc1\x83\x93\x8c\xcf\xbf\xe2\xe8\x8e83;V\xa6N[T\x1d\x14%\xce\xb3>R\xa2\xb9\x04r\x1cE0/\xe5\xb9\x1b\x03\xc8\xa3:\xdf_\xa6\xd8\xc1\x027mW\xb1x*>\x1a\x12\xe6\xdam\xe9j(=X\x938jqz\x190\x18\xe0\xdfd\xcf,\xc4\xdf\xc1G\x0b\x90\x962\xc15\xe8`\x02\x15]Y\x90\xa9\x93?\xa5K\x85\xb4\xde\x9e\xc6c\x18\xb8\xe5\xf6\xa8\xd7\xd5\xe8I\xa2\x9bU\xac\xa5H\xc3\x7f\x1d\xcd\x0c\x9e}\x00\xf8B~\xfe
Jumlah bit *chipertext* berbeda	27/256 x 100% = 10,55%
Decrypt Text	65.201250140093491

Berdasarkan tabel 4.1 dan 4.2 pengujian *Avalanche Effect* pada RSA 1024 bit menghasilkan 23,43% untuk *chipertext* nya. Sedangkan pada RSA 2048 bit menghasilkan 10,55% untuk *chipertext* nya. Karena metode RSA merupakan algoritma *block chipper* yang bersifat asimetrik sehingga tidak menghasilkan *key stream*. Pengujian *Avalanche Effect* menggunakan pergantian 1 bit pada *plaintext* RSA karena pada pembentukan kunci *public* dan *private* nya di *generate* secara acak.

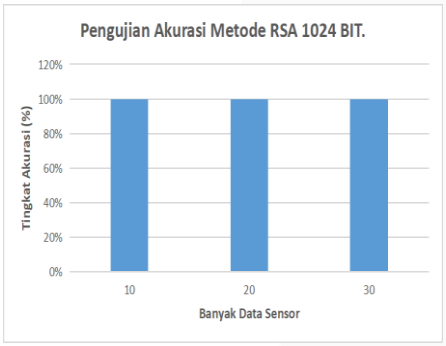
2. Pengujian Akurasi Data Sebelum di Enkripsi dan Sesudah Data di Dekripsi

Pengujian ini dilakukan untuk mendapatkan hasil dari akurasi keaslian data yang di enkripsi (*chipertext*) dengan data yang di dekripsi (*plaintext*) pada *database* menggunakan metode RSA 1024 dan 2048 bit berdasarkan jumlah data yang digunakan yaitu 10, 20, dan 30 data. Hasil dari pengujian dapat dilihat pada gambar 4,2 dan gambar 4.3. Pada tabel 4.1 dan 4.2 ditunjukkan korelasi antara *plaintext* dengan *chipertext* pada RSA 1024 dan 2048 bit.

TABEL 4. 3
TABEL KORELASI *PLAINTEXT* DENGAN *CHIPERTEXT*

RSA 1024 BIT.

Data Sensor	Enkripsi RSA 1024 Bit	Dekripsi RSA 1024 Bit
63.406240940093994	\\x04\\xcf\\x80\\xcc(O\\x82\\xf1\\x80\\x11kg\\x1a(Y~\\xc0\\xc8\\xd9\\xf1fc\\xe7\\x9f\\xee\\x85;\\xbc\\xa6\\xdebX5\\x8e\\xc5\\xa2z5\\xb7Oza\\x00\\xe3\\x98!\\xa3\\x86FX\\xd1\\xd4Ew\\x98\\xfdq\\x9a\\x90\\xf71\\x95\\x9fm\\xbfd'\\xbd\\x12H\\xa4\\xdb\\xc6\\xec\\x94\\x9eb\\x8c\\n\\xeb\t\\x87P\\xf6\\xcf'\\xb2\\x1e\\xd2\\x86\\xe4\\x95\\x02k\\xf4J\\xbb\\xbem\\xc0\\x9f\\xf7d\\xb2:b\\x1f\\x02\\xe2\\n\\xa95M3\\x15\\xf1\\x0cs\\xf5\\x9f\\x9fT\\x9126\\xc6\\x88\	63.406240940093994
60.201250140093491	L\xde\xf1k\x89\x10\x86\xe5h]\xa6\x02rm\\@\xe3\xdd\xb4\x0f\xfeR\x93\xa6\xe6\x11\xd4\xa0\xba\x11\xbdX\xa7\xb2\x00V/!\xa1\x10%fu\xae2\x13\x81\x02z\xe0\xdc\x1d\xa9\xe51\xa6\x99B\xcf7\x13\xba\x87\xe26n\xde\x1a\x86$)k_\xe4R9RNE~\x80#\x8dt=<?~\x9b\xbc\x999r\xc0\x98w\xea\x07;\xcd\xab4{\x10!\x8c\xeb\xb12\xbe8L\x9d\x91,\x1ev\xb8v\x0b#\x07\xa54*\x88\x1eL	60.201250140093491



GAMBAR 4. 2

PENGUJIAN AKURASI RSA 1024 BIT.

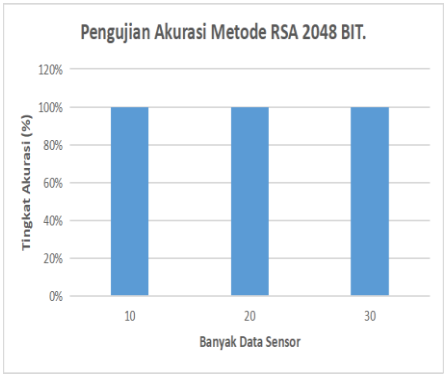
Dari hasil pengujian yang ditunjukkan pada gambar 4.2 didapatkan akurasi keaslian data yang di enkripsi (*chipertext*) dengan data yang di dekripsi (*plaintext*) pada *database* menggunakan RSA 1024 bit berdasarkan jumlah data yang digunakan yaitu 10, 20, dan 30 data sebesar 100,00%. Pada tabel 4.1 ditunjukkan korelasi antara *plaintext* dengan *chipertext* dengan menunjukkan data sensor sebelum dienkripsi dan data sensor setelah didekripsi menggunakan RSA 1024 bit.

TABEL 4. 4

TABEL KORELASI *PLAINTEXT* DENGAN *CHIPERTEXT*

RSA 2048 BIT.

Data Sensor	Enkripsi RSA 2048 Bit	Dekripsi RSA 2048 Bit
52.554250121093464	\xac\x86\xdf\xf5\xc7\x96\xa0I\xec\xea\x069\xf0\xb1N;\xbe1|H<E\xf8g\x8eJ\xe8\xa1\xb1\x8a\xee\x8a\xd3i\xc9Q\xfb\x91+Bv}D\xb2\xd2y\xecz\xae\xe3\xba\xba\xd5\xc1\xfe\x91\x15\x04\xb9IwO\x15/\xe2\xe5\xf6"vu\x9b$\xab\xd9\xfa=\xe8#:9\xf7\x9b\x94{J\xfa\xd4\x8a\x1b\x9f\xc9r=\x15S\xc6X\xd6\x1bVC\xe0\x13\x92pO\x05\x7f,\x8e(\xfbBg\x14\'\xdc\xf0\xef\xbd\xd5OROH\xefE\xdblX\xd8\xdbG\x88_\xac\x93-)\xabJ\xfb\xefzE\xa4\xd7sd\x18\xd5Yi\xf3Djc\x972]M\xf7,f\x12\x14\xca\xb9\xa6\x1fH\x1e\x94\xcc\xeeF\x86\x97\xcb\xf9\xb7\xc1\xb4\xf0\xaf\x85\x00\xe3\xc6\x19M\xc7~\xd1DZ=R&3\x9c\xcb\xfe\xa6\xa2\xd5\xcc\xe3<\xce=\x9dR\x97u\xd3\xd0\xb6\xa4<\xf8_,\xf1+{\xc9K\xc4X\x07\xa30V\xe2\xb0\xff\x97\x92\xc3p\xb5\xe7\xcdA\t7\xf7\xe6eW\xe2	52.554250121093464
58.394220221463412	Y\\xf3\\x82C\\xeb\\x8d\\xe6e\\xa5jo\\xda4]\\xb7`\\xb4\\n\\x148\\xf8\\xdf\\x0brx\\x1f\\xf8\\xbe'3c\\xfd\\xdf\\xab\\x9c@\\xe2p=\\xb6\\x82\\x0c\\xee\\x99*\\xcf\\xeb\\xd5\\x0brY\\xb7M^\\xbd3@4\\x89\\x00$\\xfat7U\\x9dL\\xb1)\\xf0!a\\x90\\r&\\xf6\\xc7\\xe4\\x1e\\xd5\\x9c\\x96\\x9e\\xf2\\xfa\\x92\\xfdu\\xaba0\\x15\\x02\\x8e\\xe3*^\\x88X\\x17D\\xd9\\x97U\\x7f&\\xb0\\xc2\\xbc;\\xff[u{w\\xb9\\x94\\xa9d\\x1c\\xf8\\xb2\\xaaaj\\xce\\xdeS\\xb4r\\xe0s\\x8e\\xbb\\xb9\\xb1\\xbfAY\\xcd(I\\x91\\xec\\t\\xafMP\\xd0;XY\\x8f\\x9a\\x85\\x7f\\x14t\\x19o\\x16r\\x90\\xd7\\x86<\\x1brtH\\x19\\xb2\\xd1\\x8c\\x0e\\xc49\\xb93\\xaf\\x9dz\\xfc\\xfcS{\\xbeN\\xf0\\xff\\x11\\x8d\\xa8\\xcc\\xf3\\x7f\\xb1\\xfaD7`X#V\\xdb\\xcdq<8(\\xa5\\x0e\\xe9yxg\\x98\\xf6\\xb3\\\\\\x89|\\x83s\\xb3\t\\xecVA5\\x94\\x9c\\xea\\xc1\\x93\\xf7\\x1fe\\x1a9A\\xff\\xa3\\x9f\\x06#\\x8d\\xc0T3\\xf4\\xd7\\xea\\xb6\	58.394220221463412



GAMBAR 4. 3
PENGUJIAN AKURASI RSA 2048 BIT.

Dari hasil pengujian yang ditunjukkan pada gambar 4.3 didapatkan akurasi keaslian data yang di enkripsi dengan data yang di dekripsi pada *database* menggunakan RSA 2048 bit berdasarkan jumlah data yang digunakan yaitu 10, 20, dan 30 data sebesar 100,00%. Pada tabel 4.2 ditunjukkan korelasi antara *plaintext* dengan *chipertext* dengan menunjukkan data sensor sebelum dienkripsi dan data sensor setelah didekripsi menggunakan RSA 2048 bit.

3. Pengujian *Response Times* Proses Enkripsi dan Dekripsi

Pada bagian ini, dilakukan pengujian *Response Times* pada sistem saat melakukan proses enkripsi dan dekripsi. Pada bagian ini dilakukan pengujian dengan menghitung *time* pada sistem dengan menggunakan skenario RSA 1024 dan 2048 bit menggunakan jumlah data sebanyak 10,20, dan 30 data.

A. Pengujian *Response Times* Proses Enkripsi dan Dekripsi RSA 1024 Bit.

TABEL 4. 5

TABEL *RESPONSE TIMES* RSA 1024 BIT.

Banyak Data Sensor	Ukuran *Plaintext (bit)*	Ukuran *Chipertext (bit)*	Waktu Enkripsi *(milisecond)*	Waktu Dekripsi *(milisecond)*
10	18	128	856,22	2075,42
20	18	128	886,08	2164,17
30	18	128	895,89	2329,9
Rata-rata			879,4	2189,83

Dari hasil pengujian *Response Times* menggunakan metode RSA 1024 bit didapatkan hasil dari pengujian enkripsi menggunakan jumlah data sebanyak 10, 20, dan 30 data dengan ukuran *plaintext* dari masing-masing *message* sebesar 72 bit dan 1024 bit untuk ukuran *chipertext* adalah 856,22 ms, 886,08 ms, dan 895,89 ms. Sedangkan *Response Times* untuk dekripsi didapatkan hasil sebesar 2075,42 ms, 2164,17 ms, dan 2329,9 ms. Untuk rata-rata hasil *Response Times* pada enkripsi RSA 1024 bit didapatkan sebesar 879,4 ms. Sedangkan rata-rata hasil *Response Times* dekripsi pada RSA 1024 bit didapatkan sebesar 2189,83 ms. Untuk hasil response times enkripsi RSA 1024 bit cukup baik karena berada di bawah 0,1 s, sedangkan untuk response times dekripsi RSA 1024 bit cukup baik karena berada di rentang nilai 1 s – 6 s .

B. Pengujian *Response Times* Proses Enkripsi dan Dekripsi RSA 2048 Bit.

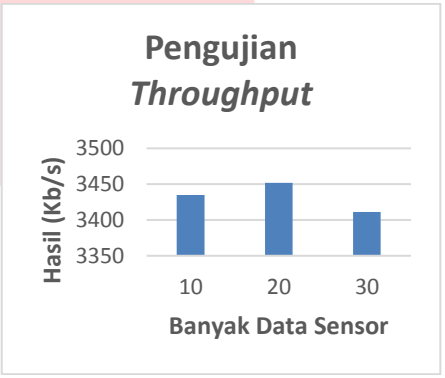
TABEL 4. 6

TABEL *RESPONSE TIMES* RSA 2048 BIT.

Banyak Data Sensor	Ukuran *Plaintext (bit)*	Ukuran *Chipertext (bit)*	Waktu Enkripsi *(milisecond)*	Waktu Dekripsi *(milisecond)*
10	18	256	4.130,58	155.213,44
20	18	256	4.447,23	230.003,81
30	18	256	4.655,33	454.178,91
Rata-rata			4.411,05	279.798,72

Dari hasil pengujian *Response Times* menggunakan metode RSA 2048 bit didapatkan hasil rata-rata dari pengujian enkripsi menggunakan sampel data sebanyak 10, 20, dan 30 data dengan ukuran *plaintext* dari masing-masing *message* sebesar 72 bit dan 2048 bit untuk ukuran *chipertext* adalah 4130,58 ms, 4447,23 ms, dan 4655,33 ms. Sedangkan *Response Times* untuk dekripsi didapatkan hasil sebesar 155213,44 ms, 230003,81 ms, dan 454178,91 ms. Untuk rata-rata hasil *Response Times* pada enkripsi RSA 2048 bit didapatkan sebesar 4411,05 ms. Sedangkan rata-rata hasil *Response Times* pada dekripsi RSA 2048 bit didapatkan sebesar 279.798,72 ms. Untuk hasil *response times* enkripsi dan dekripsi RSA 2048 bit buruk karena nilai yang didapat lebih dari 10 s.

4. Pengujian *Throughput* Sebelum Enkripsi



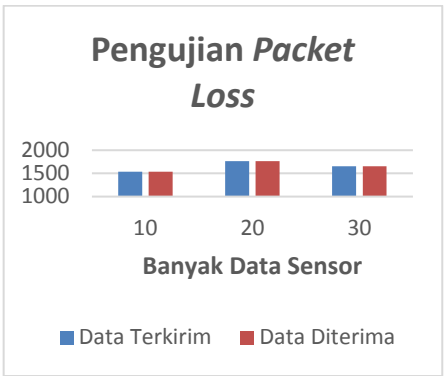
GAMBAR 4. 4

PENGUJIAN *THROUGHPUT* SEBELUM DI ENKRIPSI.

Pada gambar 4.4 menunjukkan pengujian *throughput* pada system sebelum di enkripsi. Nilai *throughput* yang didapatkan pada pengujian dengan jumlah data 10, 20, dan 30 data sebesar 3434,92 Kb/s, 3451,75 Kb/s, dan 3411,38 Kb/s. Sedangkan untuk rata-rata nilai *througput* yang didapatkan sebesar 3432,68 Kb/s.

5. Pengujian *Packet Loss* Sebelum Enkripsi

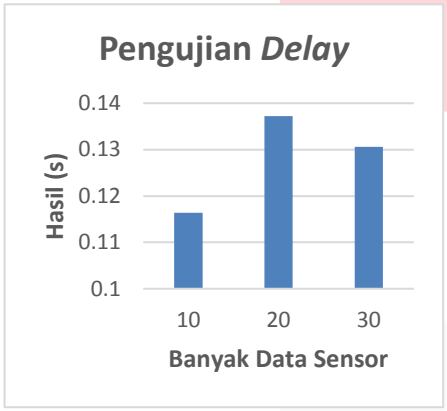
Pengujian *packet loss* dilakukan untuk pengujian tanpa menggunakan enkripsi dengan jumlah data 10, 20, dan 30 data menggunakan aplikasi Wireshark dalam 30 kali percobaan. Pengujian *packet loss* menggunakan jaringan Wi-Fi. Hasil yang diperoleh memiliki total 1.652 paket yang dikirim dan total 1.652 paket yang diterima. Dari hasil pengujian *packet loss* yang dilakukan nilai yang didapatkan sesuai menurut standar ITU-T G1010 karena tidak ada paket yang hilang.



GAMBAR 4. 5

PENGUJIAN *PACKET LOSS* SEBELUM ENKRIPSI.

6. Pengujian *Delay* Sebelum Enkripsi

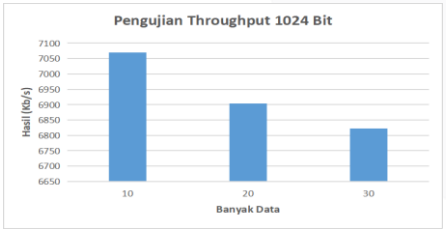


GAMBAR 4. 6

PENGUJIAN *DELAY* SEBELUM DI ENKRIPSI.

Pada gambar 4.6 menunjukkan pengujian *delay* sebelum dilakukan enkripsi. Nilai *delay* yang didapatkan pada pengujian dengan jumlah data 10, 20, dan 30 data sebesar 116,35 ms, 137,25 ms, dan 130,6 ms. Sedangkan untuk rata-rata nilai *delay* yang didapatkan sebesar 128,07 ms.

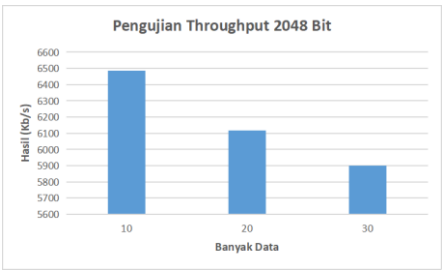
7. Pengujian *Throughput* Setelah Enkripsi



GAMBAR 4. 7

PENGUJIAN *THROUGHPUT* RSA 1024 BIT.

Pada gambar 4.7 menunjukkan pengujian *throughput* pada metode RSA 2048 bit. Nilai *throughput* yang didapatkan pada pengujian dengan jumlah data 10, 20, dan 30 data sebesar 7069,83 Kb/s, 6903,50 Kb/s, dan 6822,78 Kb/s. Sedangkan untuk rata-rata nilai *througput* yang didapatkan sebesar 6932,04 Kb/s.

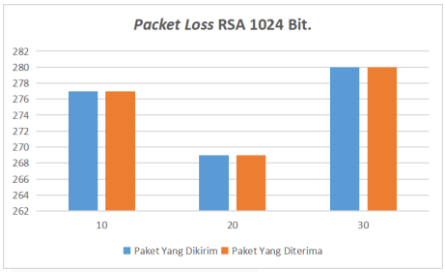


GAMBAR 4. 8

PENGUJIAN *THROUGHPUT* RSA 2048 BIT.

Pada gambar 4.8 menunjukkan pengujian *throughput* pada metode RSA 2048 bit. Nilai *throughput* yang didapatkan pada pengujian dengan jumlah data 10, 20, dan 30 data sebesar 6484,83 Kb/s, 6115,75 Kb/s, dan 5900,52 Kb/s. Sedangkan untuk rata-rata nilai *througput* yang didapatkan sebesar 6167,03 Kb/s.

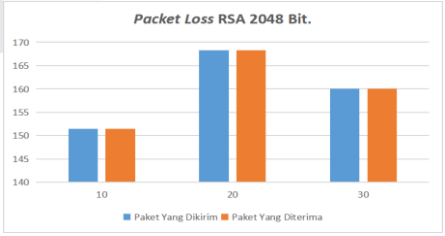
8. Pengujian *Packet Loss* Setelah Enkripsi



GAMBAR 4. 9

PENGUJIAN *PACKET LOSS* RSA 1024 BIT.

Pengujian *packet loss* dilakukan untuk pengujian skenario RSA 1024 bit dengan jumlah data 10, 20, dan 30 data menggunakan aplikasi Wireshark dalam 30 kali percobaan. Pengujian *packet loss* menggunakan jaringan Wi-Fi. Hasil yang diperoleh memiliki total 826 paket yang dikirim dan total 826 paket yang diterima. Dari hasil pengujian *packet loss* untuk skenario RSA 2048 bit yang dilakukan nilai yang didapatkan sesuai menurut standar ITU-T G1010 karena tidak ada paket yang hilang.



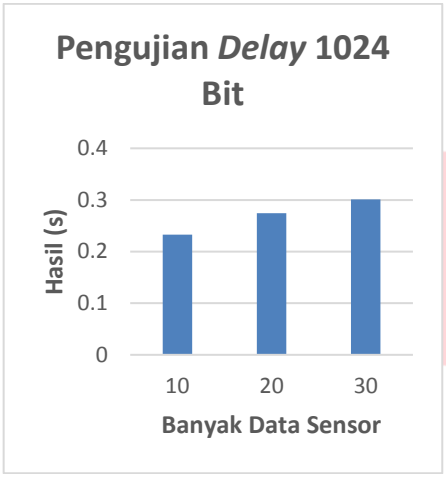
GAMBAR 4. 10

PENGUJIAN *PACKET LOSS* RSA 2048 BIT.

Pengujian *packet loss* dilakukan untuk pengujian skenario RSA 2048 bit dengan jumlah data 10, 20, dan 30 data menggunakan aplikasi Wireshark dalam 30 kali percobaan. Pengujian *packet loss* menggunakan jaringan Wi-

Fi. Hasil yang diperoleh memiliki total 480 paket yang dikirim dan total 480 paket yang diterima. Dari hasil pengujian *packet loss* untuk skenario RSA 2048 bit yang dilakukan nilai yang didapatkan sesuai menurut standar ITU-T G1010 karena tidak ada paket yang hilang.

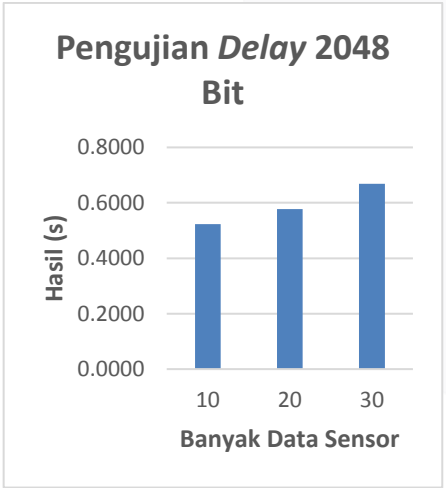
9. Pengujian *Delay* Setelah Enkripsi



GAMBAR 4. 11

PENGUJIAN *DELAY* RSA 1024 BIT.

Pada gambar 4.11 menunjukkan pengujian *delay* pada metode RSA 1024 bit. Nilai *delay* yang didapatkan pada
pengujian dengan jumlah data 10, 20, dan 30 data sebesar 205,4 ms, 274,5 ms, dan 301,2 ms. Sedangkan untuk rata-rata nilai *delay* yang didapatkan sebesar 243,3 ms.



GAMBAR 4. 12

PENGUJIAN *DELAY* RSA 2048 BIT.

Pada gambar 4.12 menunjukkan pengujian *delay* pada metode RSA 2048 bit. Nilai *delay* yang didapatkan pada pengujian dengan jumlah data 10, 20, dan 30 data sebesar 522,8 ms, 577,5 ms, dan 668,4 ms. Sedangkan untuk rata-rata nilai *delay* yang didapatkan sebesar 589,5 ms. Dari hasil pengujian *delay* untuk skenario RSA 1024 bit nilai yang didapatkan dibawah 30 s dan 250 ms sehingga sesuai dengan standar ITU-T G1010. Sedangkan RSA 2048 bit melebihi 250 ms.

V. KESIMPULAN DAN SARAN

A. Kesimpulan

Setelah dilakukannya pengujian hasil enkripsi didalam protokol MQTT dan dekripsi didalam *database* PHP *MyAdmin*, dapat disimpulkan bahwa:

1. Sistem yang telah dibuat dapat melakukan enkripsi dan dekripsi menggunakan metode RSA pada data sensor ultrasonik HC-SR04.
2. Sistem mampu melakukan *transfer* data sensor melalui protokol MQTT kedalam *database* di PHP *MyAdmin* dan proses dekripsi dilakukan secara otomatis didalam *database* di PHP *MyAdmin*.
3. Pengujian *Avalanche Effect* pada RSA 1024 bit menghasilkan 23,43% untuk *chipertext* nya. Sedangkan pada RSA 2048 bit menghasilkan 10,55% untuk *chipertext* nya.
4. Akurasi keaslian data yang di enkripsi (*chipertext*) dengan data yang di dekripsi (*plaintext*) pada *database* menggunakan RSA 1024 dan 2048 bit berdasarkan jumlah data yang digunakan yaitu 10, 20, dan 30 data sebesar 100,00%.
5. Hasil rata-rata *Response Times* untuk skenario RSA 1024 bit sebesar 879,4 ms untuk proses enkripsi dan 2189,83 ms untuk proses dekripsi. Untuk skenario RSA 2048 bit didapatkan hasil rata-rata pengujian *Response Times* untuk 10, 20, dan 30 data sebesar 2.859,27 ms untuk proses enkripsi dan 228.060,91 ms untuk proses dekripsi.
6. Pengujian *throughput* menggunakan jaringan Wi-Fi untuk sistem yang dilakukan sebelum enkripsi diperoleh hasil sebesar 3432,68 Kb/s. Sedangkan pengujian *throughput* Fi untuk metode RSA 1024 dan 2048 bit diperoleh hasil sebesar 6932,04 Kb/s dan 6167,03 Kb/s. Hasil *throughput* bersifat tidak stabil, tergantung dari kualitas jaringan, banyaknya pengguna jaringan, dan jenis jaringan yang dipakai.
7. Pada pengujian *packet loss* sebelum enkripsi menggunakan jaringan Wi-Fi didapatkan hasil sebesar 0%. Sedangkan untuk pengujian *packet loss* menggunakan jaringan Wi-Fi didapatkan hasil sebesar 0% untuk skenario RSA 1024 bit dan RSA 2048 bit yang dilakukan, yang berarti semua data berhasil terkirim. Dari hasil pengujian *packet loss* yang dilakukan nilai yang didapatkan sesuai menurut standar ITU-T G1010 karena tidak ada paket yang hilang.
8. Pada hasil pengujian *delay* menggunakan jaringan Wi-Fi untuk pengujian sebelum enkripsi didapatkan hasil *delay* rata-rata sebesar 128,07 ms. Sedangkan Pengujian *delay* menggunakan jaringan Wi-Fi untuk skenario RSA 1024 dan 2048 bit didapatkan hasil *delay* rata-rata sebesar 243,3 ms dan 589,5 ms. Dari hasil pengujian *delay* yang dilakukan nilai yang didapatkan dibawah 30 s dan 250 ms sehingga sesuai dengan standar ITU-T G1010. Sedangkan RSA 2048 bit melebihi 250 ms.

B. Saran

Peneliti menyadari hasil dari implementasi, pengujian, dan penulisan tugas akhir ini masih jauh dari kata sempurna. Maka terdapat beberapa kekurangan yang bisa dijadikan saran untuk pengembangan penelitian selanjutnya, yaitu :

1. Menggunakan metode enkripsi data yang berbeda yang diharapkan lebih aman untuk menjaga data IoT.
2. Menambahkan metode pengamanan data lain agar dapat dikolaborasikan menjadi pengamanan ekstra.
3. Melakukan pengujian serangan terhadap sistem keamanan data yang telah dibuat.

REFERENSI

[1] Fitri Febriyan C. 2018. PROTOTIPE SISTEM ENKRIPSI DAN DEKRIPSI BERBASIS FPGA MENGGUNAKAN ALGORITMA STREAM CIPHER TRIVIUM. Bandung: Universitas Telkom, S1 Sistem Komputer.

[2] F. P. Aji, A. Solehudin, and C. Rozikin, "Implementasi sensor ultrasonik dalam mendeteksi volume limbah b3 pada tempat sampah berbasis internet of things," Jurnal Ilmiah Informatika, vol. 6, no. 2, pp. 117–126, 2021.

[3] Chandra Putra Devha. 2013. Pengamanan Pesan Rahasia Menggunakan Algoritma Kriptografi Rivest Shank Adleman (RSA). (online). (respository.upi.edu/2939/6/SMTK 0905803 CHAPTER3.pdf)

[4] Hudan A.R., Rakhmadhany P., Heru N., 2017. Sistem Kendali Berbasis Mikrokontroler Menggunakan Protokol MQTT pada Smarthome. Malang: Universitas Brawijaya.

[5] R. Munir, Kriptografi. Bandung: Informatika, 2016

[6] Ardi. 2017. Perbandingan Algoritma Massage Digest 5 (MD5) dan SHA256 Pada Hashing File Dokumen. Sumatera Utara: Universitas Sumatera Utara.

[7] Angga, Christian. 2011. Analisis Cara Kerja Beragam Fungsi Hash yang Ada.Bandung: Institut Teknologi Bandung.

[8] Butler, T & Yank, K. PHP & MySQL: Novice to Ninja, 6th Edition. SiitePoing. 2016.

[9] Solihin,A. 2015. MySQL Dari Pemula Hingga Mahir [Internet]dihttps://www.researchgate.net/publication/236885803_MySql_5_Dari_Pemula_Hingga_Mahir.

[10] Nusantara, M. F., Akbar, S. R. & Rachmadi, A., 2016. Analisa Metode *Publish/Subcribe* untuk Komunikasi Antar Perangkat Dalam Lingkungan *Smarthome*.

[11] Lampin, V. et al., 2012. Building Smarter Planet Solutions with *MQTT* and IBM.

[12] Ariyus, Doni. Pengantar Ilmu Kriptografi Teori dan Implementasi. Andi Offset : Yogyakarta. 2008.

[13] Komputer, Wahana. The Encryption Tools. PT Elexmedia Komputindo : Jakarta. 2010.

[14] Brink, J. van den (Pieter J. C. M., Monumentenstichting Baet en Borgh., N. B., Historische Vereniging Tweestromenland., J., & Azanuddin, A. (2014). Mensen van Maas en Waal. Jurnal SAINTIKOM (Jurnal Sains Manajemen Informatika Dan Komputer), 18(1), 30–34. https://ojs.trigunadharma.ac.id/index.php/j is/article/view/100

[15] Ginting, A., Isnanto, R. R., & Windasari, I. P. (2015). Implementasi Algoritma Kriptografi RSA untuk Enkripsi dan Dekripsi Email. Jurnal Teknologi Dan Sistem Komputer, 3(2), 253. https://doi.org/10.14710/jtsiskom.3.2.2015 .253-258

[16] Zhao, S. C. Exploring IOT Application Using Raspberry Pi . *International Journal of Computer Networks and Applications.* 2015.

[17] ITU-T, "G.1010: End-user multimedia QoS categories," Int. Telecommun.Union, vol. 1010, 2001, [Online].

[18] Ferry, "Response Time Testing," – School of Information Systems (binus.ac.id), 2019, [Online].