Sistem Pengamanan Data IoT Menggunakan Enkripsi AES

1st Putri Stri Wahyuni
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
putrisriwahyuni@student.telkomuniversity.ac.id

2nd Muhammad Ary Murti
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
arymurti@telkomuniversity.ac.id

3rd Gandeva Bayu Satrya
Fakultas Ilmu Terapan
Universitas Telkom
Bandung, Indonesia
gbs@telkomuniversity.ac.id

Abstrak—*Internet of Things* (IoT) telah memberi kontribusi besar terhadap perkembangan teknologi komunikasi dalam kehidupan sehari-hari kita. Kemudahan untuk mengadopsi teknologi ini diperkuat oleh bertambahnya jumlah perangkat elektronik yang saling terhubung melalui internet seperti yang ditunjukkan oleh hasil-hasil riset sebelumnya. Namun besarnya pertumbuhan jumlah penggunaan pada sistem ini seharusnya seiring dengan besarnya kepedulian untuk melakukan tindakan pengamanan pada komunikasi sistem IoT. Sebagai contoh, kejahatan siber bisa saja dilakukan terhadap sistem IoT yang tidak menggunakan topologi dan protokol yang benar, atau data yang dikirim dari perangkat sensor IoT tidak dilindungi dengan semestinya. Penelitian ini mengusulkan sebuah jalur komunikasi aman antara perangkat sensor IoT ke Internet. Penelitian ini mendemonstrasikan protokol komunikasi dengan sistem enkripsi AES.

Kata Kunci—*IoT,* enkripsi *AES*, mikrokomputer, keamanan siber.

I. PENDAHULUAN

Paradigma *Internet of Things* (IoT) mengacu pada jaringan objek fisik atau "benda" yang disematkan dengan elektronik, perangkat lunak, sensor, dan konektivitas untuk memungkinkan objek bertukar data dengan server, sistem terpusat, dan/atau perangkat terhubung lainnya berdasarkan ragam infrastruktur komunikasi [1]. Dalam siaran pers November 2014, Gartner, Inc. memperkirakan bahwa pada 2015 akan ada 4,9 miliar perangkat IoT yang digunakan di seluruh dunia tumbuh menjadi 25 miliar perangkat pada tahun 2020 [2]. *Internet of Everything* (IoE) tidak terhubung untuk menciptakan nilai bisnis misalnya, orang, data, proses, dan lainnya [3]. Perangkat fisik dan hal-hal yang terhubung ke Internet dan satu sama lain untuk pengambilan keputusan yang cerdas juga disebut IoT. Jadi, IoT mengubah data menjadi pengalaman, yaitu informasi data penting melalui jaringan publik, data besar, dan lain-lain. Dalam IoT, keamanan informasi data melalui Internet harus ditangani dengan hati-hati karena saat ini terdapat sekitar 50 miliar objek pintar yang telah terhubung [4].

Penerapan sistem IoT digunakan untuk memberikan solusi pada berbagai macam masalah. Keamanan data dan privasi tentunya merupakan salah satu permasalahan utama yang dihadapi oleh semua organisasi dan perusahaan yang terhubung ke internet. Meskipun IoT memiliki begitu banyak potensi bagi perkembangan dunia digital, tetapi dalam penerapannya, sistem IoT masih memiliki beberapa masalah utama seperti heterogenitas perangkat, identitas perangkat, manajemen perangkat, dan pengamanan komunikasi perangkat. Kemungkinan terjadinya kejahatan siber pada perangkat IoT dimungkinkan karena banyaknya data pribadi dan penting yang dikumpulkan pada perangkat IoT yang bisa dimanfaatkan untuk tujuan jahat. Oleh karena itu, memfasilitasi sistem dan protokol komunikasi yang aman bagi konsumen menjadi tanggung jawab bagi vendor-vendor perangkat sistem IoT.

Untuk membuat suatu sistem IoT dibutuhkan sensor, perangkat mikrokontroler, server, dan jaringan nirkabel yang menggunakan protokol komunikasi berbeda. Protokol komunikasi pada IoT yang paling umum digunakan adalah MQTT. MQTT merupakan protokol paling umum yang digunakan sebagai standar protokol komunikasi untuk sistem IoT karena dengan menggunakan protokol MQTT tidak diperlukan banyak protokol tambahan dan kemampuannya untuk melakukan pengiriman data dengan cepat. Penelitian yang dilakukan Goyena R. dan Fallis A. menunjukkan bahwa celah keamanan yang membolehkan penyerang untuk melakukan pencurian informasi dari perangkat IoT dapat memberikan dampak yang serius. Alasan tersebut membawa penelitian ini dengan tujuan untuk mengoptimalkan penerapan protokol MQTT pada lingkungan IoT tanpa mengorbankan banyak sumber daya. Dengan mengunakan topologi ini peneliti menerapkan sistem pengamanan data yang dikirim oleh perangkat IoT ke server atau data *center*.

Pada tahun 2001, NIST (*National Institute of Standard and Technology*) mempublikasikan algoritma enkripsi data baru untuk menggantikan algoritma DES (*Data Encryption Standard*) yang memiliki beberapa kelemahan. Algorima baru ini dinamakan AES (*Advanced Encryption Standard*) atau Rijndael [19]. Saat ini, AES merupakan algoritma *cipher* yang cukup aman untuk melindungi data atau informasi yang bersifat rahasia. Algoritma AES adalah algoritma kriptografi yang dapat mengenkripsi dan mendekripsi data dengan panjang kunci yang bervariasi, yaitu 128 bit, 192 bit, dan 256 bit [18].

II. KAJIAN TEORI

A. Enkripsi AES

Algoritma AES termasuk pada algoritma kriptografi modern simetris dengan menggunakan blok ukuran tertentu dalam proses enkripsi dan dekripsinya. Algoritma AES ini menggunakan kunci kriptografi 128, 192, dan 256 bit untuk mengenkripsi dan dekripsi data pada blok 128 bit [7].

Kelebihan Enkripsi AES [20]:

1. Dilihat dari segi jenis kunci yang simetris, maka kecepatan operasi (komputasi) lebih tinggi bila dibandingkan dengan algoritma asimetris sehingga dapat digunakan pada sistem *realtime*.
2. AES mempunyai panjang kunci paling sedikit 128 bit, maka AES tahan terhadap serangan *exhaustive key search* dengan teknologi saat ini. Dengan panjang kunci 128-bit, maka terdapat $2^{128} \approx 3{,}4 \times 10^{38}$ kemungkinan kunci.
3. Operasi matematis yang kompleks dan membutuhkan sumber daya yang tidak sedikit untuk melakukan komputasi.

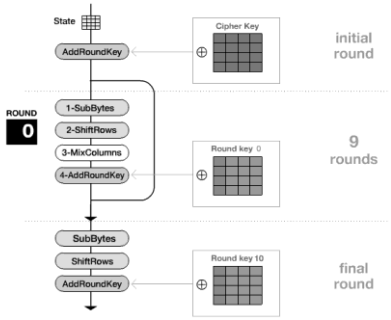
Kekurangan Enkripsi AES [20]:

1. Dari segi jenis kunci yang simetris, maka akan terjadi kesulitan dalam manajemen kunci. Hal ini terjadi karena untuk setiap pengiriman dan penerimaan data dengan pengguna yang berbeda dibutuhkan kunci yang berbeda pula.
2. Pengirim dan penerima data memiliki kunci yang sama untuk setiap proses pengiriman-penerimaan data, hal ini akan menyebabkan kunci mudah bocor meskipun dalam waktu yang lama.
3. Dengan berhasilnya dipecahkan persamaan matematis yang mendasarinya secara otomatis seluruh sistem di dalam AES dapat ditembus dan dengan demikian barisan pertahanannya dapat dikatakan hancur berantakan.

Proses enkripsi algoritma AES terdiri dari empat jenis transformasi *bytes*, yaitu *SubBytes*, *ShiftRows*, *Mixcolumns*, dan *AddRoundKey*. Sedangkan untuk tahapan prosesnya sebagai berikut:

1. Menentukan *plain text* dan *key* yang akan digunakan.
2. Berdasarkan *plain text* yang sudah ditentukan dibuatlah blok *plain text* berukuran 4x4 yang disubtitusi dengan tabel ASCII (*American Standard Code for Information Interchange*). Begitupula dengan *key* yang telah ditentukan, diubah ke blok bilangan heksadesimal berukuran 4x4, lalu disubtitusi berdasarkan tabel ASCII.
3. Menentukan *initial round*, yang didapatkan dari *plain text* dixor dengan *key*.
4. Menentukan *generate key* berdasarkan *initial round.* Pada proses ini akan membangkitkan 10 buah kunci karena pada proses enkripsi tiap *round* ada proses *add round key* yang memiliki 10 *round*.
5. Dilakukan tahapan *subbytes, shiftrows, mixcolumn, add round key*.
6. Mengulangi tahap (5) untuk *round*-2 sampai dengan *round*-9.
7. Pada *round*-10 tetap melakukan sesuai tahap (5), tetapi tidak melakukan proses *mixcolumn*.
8. Dari hasil *round*-10 didapatkan nilai *cipher text* dari enkripsi AES.

Untuk ilustrasi proses enkripsi AES dijelaskan pada Gambar 1.



GAMBAR 1
ILUSTRASI PROSES AES [18]

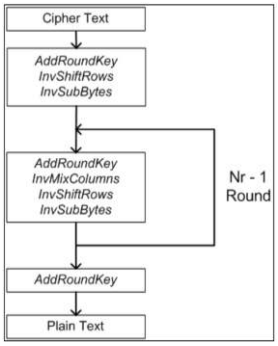
B. Proses Dekripsi

Transformasi *cipher* dapat dibalikkan dan diimplementasikan dalam arah yang berlawanan untuk menghasilkan *inverse cipher* yang mudah dipahami untuk algoritma AES. Transformasi *byte* yang digunakan pada *invers cipher* adalah *InvShiftRows*, *InvSubBytes*, *InvMixColumns*, dan *AddRoundKey* [18].

Untuk tahapan proses pengerjaannya sama dengan enkripsi, yaitu:

1. Melakukan *invers addroundkey*. Dimana *cipher text* dixor dengan *roundkey round*-10 pada proses enkripsi sebelumnya.
2. Dari hasil *invers addroundkey* diproses kembali melalui *round*-10 seperti proses enkripsi.
3. Tahap dekripsi: *invers shiftrow, invers subbyte, invers, addroundkey,* dan *invers mixcolumn*.
4. Pada *round* paling akhir tidak menyertakan tahap *mixcolumn*. Dari hasil ini, didapatkan nilai *plain text*.

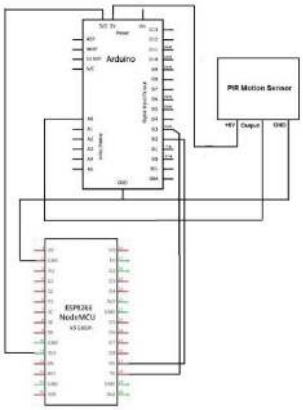
Algoritma dekripsi dapat dilihat pada skema Gambar 2:



GAMBAR 2
ILUSTRASI PROSES DEKRIPSI AES [18]

III. METODE

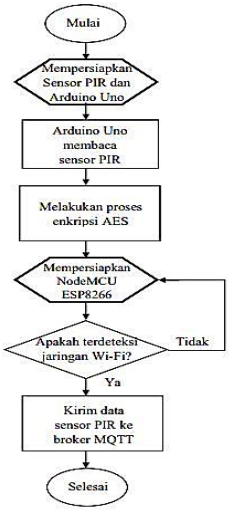
A. Desain Perangkat Keras



GAMBAR 3
DESAIN PERANGKAT KERAS

Komponen yang digunakan: Arduino Uno, Sensor PIR, dan NodeMCU ESP8266.

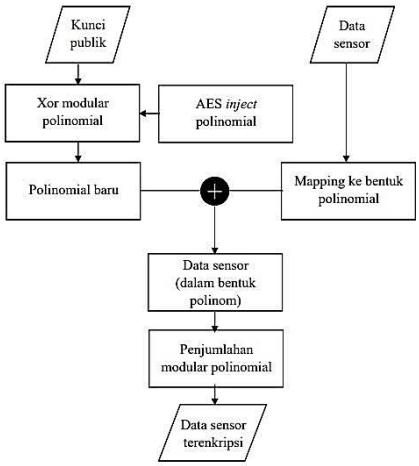
B. Diagram Alir Proses dari Sensor ke MQTT



GAMBAR 4
DIAGRAM ALIR RANGKAIAN

Pada Gambar 4 merupakan alur kerja rangkaian alat sistem pengamanan data IoT dimana sistem bekerja dimulai dengan mempersiapkan Sensor dan Arduino kemudian Arduino membaca sensor PIR serta mempersiapkan NodeMCU jika jaringan terdeteksi akan lanjut ke pengiriman data pada sensor PIR ke perangkat broker MQTT.

C. Diagram Alir Enkripsi AES

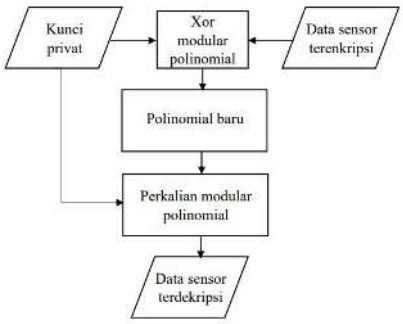


GAMBAR 5
DIAGRAM ALIR ENKRIPSI DATA

Pada Gambar 5 merupakan alur kerja enkripsi data sistem pengamanan data IoT dimana sistem bekerja dimulai dengan sensor membaca data kemudian dialgoritma dengan xor *modular polynomial* kemudian lanjut ke penjumlahan *modular polynomial* dan data sudah terenkripsi.

Proses enkripsi pesan pada AES membutuhkan kunci publik dan juga sebuah polinomial AES secara acak. Kunci publik pada awalnya akan dikalikan dengan polinom acak hingga menghasilkan polinom baru. Polinom baru tersebut kemudian dijumlahkan dengan pesan yang sudah di*mapping* ke dalam bentuk polinom yaitu berupa data sensor.

D. Diagram Alir Dekripsi AES



GAMBAR 6
DIAGRAM ALIR DEKRIPSI DATA

Pada Gambar 6 merupakan alur kerja dekripsi data sistem pengamanan data IoT dimana sistem bekerja dimulai dengan pembacaan data yang sudah terenkripsi kemudian dialgoritma dengan xor *modular polynomial* kemudian lanjut ke perkalian *modular polynomial* dan data sudah didekripsi.

Proses dekripsi pesan pada AES membutuhkan kunci privat dan juga sebuah polinomial yang merupakan hasil *invers* modulo dari kunci privat

dengan bilangan polinomial. Kunci privat pada awalnya akan di-XOR dengan pesan yang terenkripsi, sehingga membentuk polinom baru. Polinom baru tersebut akan dikalikan dengan kunci privat sehingga diperoleh pesan yang sudah terdekripsi.

IV. HASIL DAN PEMBAHASAN

A. Hasil Percobaan

Pengujian untuk mengukur konsumsi waktu enkripsi yang dilakukan pada perangkat IoT Arduino Uno, sedangkan pengujian untuk mengukur kehandalan enkripsi saat diimplementasikan pada protokol MQTT dilakukan pada perangkat *publisher* Arduino Uno, perangkat broker Arduino Uno, dan perangkat *subscriber* sebuah *desktop*.

B. Hasil Pengujian Enkripsi AES

TABEL 1
DATA HASIL PENGUJIAN ENKRIPSI

Parameter	Data Dikirimkan	Data Diterima	Waktu Pengiriman Data (detik)
	Sebelum Enkripsi	Sesudah Enkripsi	
Pesan	0000	qrst	5.39
	0058	qrv|	7.76
	0000	qrst	8.07
	0000	qrst	8.08
	0058	qrv|	10.32
	0046	qrwr	7.91
	0000	qrst	5.86
	0000	qrst	8.1
	0000	qrst	8.32
	0000	qrst	9.01
	0059	qrv}	7.06
	0000	qrst	7.84
Rata – Rata			7.81

Berdasarkan hasil pengujian enkripsi pada Tabel 1 dapat dilihat bahwa pada percobaan enkripsi tidak terjadi *error*. Didapatkan dari hasil percobaan bahwa rata – rata waktu kecepatan pengiriman data enkripsi yaitu 7.81 detik.

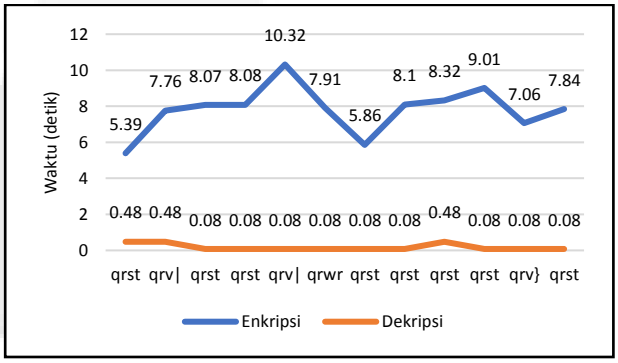
C. Hasil Pengujian Dekripsi AES

TABEL 2
DATA HASIL PENGUJIAN DEKRIPSI

Parameter	Data Dikirimkan	Data Diterima		Waktu Pengiriman Data (detik)
	Sebelum Enkripsi	Sesudah Enkripsi	Sesudah Dekripsi	
Pesan	0000	qrst	0000	0.48
	0058	qrv|	0058	0.48
	0000	qrst	0000	0.8
	0000	qrst	0000	0.8
	0058	qrv|	0058	0.8
	0046	qrwr	0046	0.8
	0000	qrst	0000	0.8
	0000	qrst	0000	0.8
	0000	qrst	0000	0.48
	0000	qrst	0000	0.8
	0059	qrv}	0059	0.8
	0000	qrst	0000	0.8
Rata – Rata				0.18

Berdasarkan hasil pengujian enkripsi pada Tabel 2 dapat dilihat bahwa pada percobaan dekripsi tidak terjadi *error*. Didapatkan dari hasil percobaan bahwa rata – rata waktu kecepatan pengiriman data enkripsi yaitu 0.18 detik.

D. Analisis Data Enkripsi dan Dekripsi



GAMBAR 7
GRAFIK WAKTU PROSES ENKRIPSI DAN DEKRIPSI

Terdapat perbedaan waktu kecepatan pada proses enkripsi dan dekripsi, hal ini lantaran disebabkan oleh jaringan internet dan *resource* yang digunakan. Dimana proses enkripsi terjadi di alat (sensor dan Arduino Uno), sedangkan proses dekripsi terjadi di alat (Arduino Uno dan NodeMCU ESP8266) ke MQTT dan *web*.

V. KESIMPULAN

Kesimpulan yang dapat diambil dari hasil pengujian dan analisis yang telah dilakukan, yaitu: sistem pengiriman data IoT melalui protokol MQTT dengan perangkat Arduino Uno bisa direalisasikan. Serta sistem enkripsi AES dapat melakukan enkripsi pada pesan yang akan dikirim melalui protokol MQTT. Sistem pengamanan data pada sistem IoT menggunakan metode enkripsi AES membuktikan bahwa penggunaan sistem enkripsi dengan tingkat keamanan tinggi bisa diterapkan pada perangkat IoT yang murah.

REFERENSI

[1] E. Bertino, "Data security and privacy in the IoT," 2016, doi: 10.5441/002/edbt.2016.02.

[2] L. c. Miller, *DDoS for Dummies*. 2012.

[3] M. Geller and P. Nair, "5G Security Innovation with Cisco," *Whitepaper Cisco Public*, 2018.

[4] Y. M. Agus, M. D. Falih, and G. B. Satrya, "On the possibilities of cybercrime in IoT devices," *Test Eng. Manag.*, 2020.

[5] R. T. Fielding, J. Gettys, J. C. Mogul, H. F. Nielsen, and T. Berners-Lee, "RFC 2068: Hypertext Transfer Protocol - HTTP/1.1," *IETF Networking Group*, 1997. .

[6] P. Saint-Andre, "RFC 6121: Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence," *Internet Eng. Task Force*, 2011.

[7] A. Banks and R. Gupta, "MQTT Version 3.1.1 Edited by Andrew Banks and Rahul Gupta. 29 October 2014. OASIS Standard.," *OASIS Stand.*, 2014.

[8] J. O'Hara, "Toward a commodity enterprise middleware," *Queue*. 2007, doi: 10.1145/1255421.1255424.

[9] C. Bormann, A. P. Castellani, and Z. Shelby, "CoAP: An application protocol for billions of tiny internet nodes," *IEEE Internet Comput.*, 2012, doi: 10.1109/MIC.2012.29.

[10] R. Goyena and A. . Fallis, "MQTT Essentials - A Lightweight IoT Protocol," *Journal of Chemical Information and Modeling*. 2019.

[11] M. Liyanage, A. Braeken, P. Kumar, and M. Ylianttila, *IoT Security: Advances in Authentication*. 2019.

[12] J. Hoffstein, J. Pipher, and J. H. Silverman, "{{NTRU}:} a new high speed public key cryptosystem," 1998.

[13] M. Singh, M. A. Rajan, V. L. Shivraj, and P. Balamuralidhar, "Secure MQTT for Internet of Things (IoT)," 2015, doi: 10.1109/CSNT.2015.16.

[14] D. Dragomir, L. Gheorghe, S. Costea, and A. Radovici, "A survey on secure communication protocols for IoT systems," 2017, doi: 10.1109/SIoT.2016.012.

[15] S. Andy, B. Rahardjo, and B. Hanindhito, "Attack scenarios and security analysis of MQTT communication protocol in IoT system," 2017, doi: 10.1109/EECSI.2017.8239179.

[16] K. Grgić, I. Špeh, and I. Hedi, "A web-based IoT solution for monitoring data using MQTT protocol," 2016, doi: 10.1109/SST.2016.7765668.

[17] J. A. Buchmann, E. Karatsiolis, and A. Wiesmaier, *Introduction to public key infrastructures*. 2013.

[18] V. Yuniati, G. Indriyanta, and A. Rachmat, "Enkripsi dan Dekripsi dengan Algoritma AES 256 untuk Semua Jenis File", 2009.

[19] Adiwidya B. M. D, "Algortima AES (*Advanced Encryption Standard*) dan Penggunaannya dalam Penyandian Pengompresian Data", 2009.

[20] Asriyanik, "Studi Terhadap *Advanced Encryption Standard* (AES) dan Algoritma *Knapsack* dalam Pengamanan Data", 2017.