e-ISSN 2549-7472
Edumatic: Jurnal Pendidikan Informatika
Vol. 5 No. 2, Desember, 2021, **Hal. 252-260**
DOI: 10.29408/edumatic.v5i2.4113

Pengamanan *Internet of Things* Berbasis NodeMCU Menggunakan Algoritma AES pada Arsitektur *Web Service* REST

Ariyan Zubaidi[*1], Rhomy Idris Sardi [2], Andy Hidayat Jatmika[3]
[1,2,3] Program Studi Teknik Informatika, Universitas Mataram
email: zubaidi13@unram.ac.id*[1], rhomyidrissardi@gmail.com[2], andy@unram.ac.id[3]

(Received: 21 Oktober 2021 / Accepted: 18 November 2021 / Published Online: 20 Desember 2021)

Abstrak

Isu keamanan data dan terbatasnya sumberdaya merupakan tantangan yang dihadapi oleh teknologi *Internet of Things*. Untuk dapat mengimplementasikan keamanan yang baik pada sistem IoT, kriptografi dapat diimplementasikan, tetapi perlu dipilih algoritma enkripsi yang efektif dan juga tidak membutuhkan banyak sumberdaya. Tujuan penelitian ini yaitu mengamankan sistem IoT dengan mengimplementasikan algoritma yang efektif dalam menjaga kerahasiaan data pada perangkat yang memiliki sumberdaya yang terbatas. Metode penelitian ini menggunakan pendekatan eksperimental, yaitu dengan membuat suatu sistem IoT untuk pertanian dan menambahkan algoritma enkripsi untuk merubah data yang ditransmisikan. Sistem IoT menggunakan NodeMCU sebagai mikrokontroler. NodeMCU merupakan mikrokontroler dengan *resources* yang kecil sehingga perlu algoritma yang efisien untuk diimplementasikan pada NodeMCU. Salah satu algoritma yang memiliki kinerja yang baik pada lingkungan komputasi *desktop* adalah algoritma kriptografi *Advance Encryption Standard* (AES). Algoritma dicoba pada lingkungan komputasi IoT dengan arsitektur pertukaran data menggunakan *web service* REST (*Representational State Transfer*), sehingga menghasilkan sistem IoT untuk pertanian dengan implementasi kriptografi di dalamnya. Pada pengujian yang dilakukan, proses enkripsi 128 dan 256 bit *plain text* membutuhkan waktu 266,31 dan 274,31 mikrodetik, sedangkan memori yang digunakan sebesar 16 % dan 17% dari total memori. Ini menunjukkan waktu enkripsi relatif cepat dan penggunaan memori relatif kecil.

Kata kunci: *Advanced Encryption Standard*, *Internet of Things*, Kriptografi, NodeMCU, REST

Abstract

Data confidentiality and resource's limitation issues are challenges for the Internet of Things. To implement good security on IoT systems, cryptography can do it, but it needs an effective encryption algorithm that does not require a lot of resources. The purpose of this study is to secure an IoT system by implementing an algorithm that is successful in maintaining the confidentiality of data transmitted. This research uses an experimental approach, by creating an IoT system for agriculture and adding an encryption algorithm. The IoT system uses NodeMCU as a microcontroller. NodeMCU is a microcontroller with small resources so it needs an efficient algorithm to be implemented in it. One algorithm that has good performance in a desktop computing environment is the Advance Encryption Standard (AES) algorithm. The algorithm is tested in an IoT computing environment with a data exchange architecture using an REST (Representational State Transfer) web service, resulting in an IoT system for agriculture with cryptographic implementations in it. In the tests carried out, the encryption process of 128 and 256 bits of plain text took 266.31 and 274.31 microseconds, while the memory used was 16% and 17% of the total memory, respectively. This shows the encryption time is fast, and the memory usage is relatively small.

Keywords: *Advanced Encryption Standard, Cryptography, Internet of Things, NodeMCU, REST*

PENDAHULUAN

Salah satu aspek penting dalam pengembangan *Internet of Things* (IoT) adalah keamanan (Ravida & Santoso, 2020) (Meutia, 2015). Tanpa fondasi keamanan yang kuat,

serangan dan malfungsi dapat lebih sering terjadi daripada manfaat yang didapatkan (Roman et al., 2017). Contoh gangguan keamanan yang terjadi misalnya *sniffing* yang dapat dilakukan dengan tool *Wireshark* pada IoT tanpa pengamanan (Joyoputro et al., 2018), dan peretasan pada *platform smart home* (Ravida & Santoso, 2020). Berbagai bentuk gangguan lainnya pada sistem IoT seperti *Denial of Service* (DoS), kerusakan fisik, *eavesdropping, node capture*, dan pengendalian (Roman et al., 2013). IoT yang berjalan dengan transmisi data melalui jaringan tentu saja lebih rawan, sehingga keamanan jaringan sangat diperlukan (Wahyudi & Utomo, 2021).

Salah satu cara efektif dalam pengamanan sistem IoT adalah dengan implementasi kriptografi. Kriptografi merupakan perlindungan terhadap informasi dan komunikasi melalui penggunaan kode sehingga hanya dapat dibaca dan diproses oleh orang yang berwenang (Bhandari & Kirubanand, 2019). Pada kriptografi, akan dilakukan proses enkripsi untuk merubah informasi menjadi bentuk yang tidak bisa dibaca dengan fungsi tertentu. Menurut dokumen IEC (*International Electrotechnical Commision*) nomor 62591, enkripsi data merupakan unsur penting pada *Internet of Things* (Roman et al., 2017).

Selain aspek keamanan, keterbatasan kemampuan komputasi juga merupakan isu yang harus diperhatikan pada perangkat IoT. Perangkat IoT biasanya memiliki kemampuan pemrosesan yang terbatas (Skirelis & Navakauskas, 2017), terdapat peningkatan kemampuan tetapi masih tidak dapat bersaing dengan komputasi *desktop* (Babar & Sohail Khan, 2021). Terbatasnya energi dan kemampuan komputasi merupakan tantangan besar pada IoT (Wei et al., 2020). Dalam melakukan komputasi pada perangkat IoT, perlu memperhatikan kemampuan komputasi sehingga didapatkan eksekusi yang efektif dan efisien.

Agar dapat memberikan keamanan pada sistem IoT maka diperlukan kriptografi untuk mengacak informasi yang diproses sehingga tidak bisa dibaca oleh selain penerima yang berhak. Pemilihan algoritma enkripsi juga perlu disesuaikan dengan kemampuan komputasi yang terbatas pada perangkat IoT. Pada penelitian yang dilakukan oleh (Asang & Sembiring, 2017), dilakukan implementasi algoritma Rivest-Shamir-Adleman (RSA) pada perangkat Arduino Uno. Algoritma dapat diimplementasikan tetapi dengan konsumsi memori yang tidak sedikit. Penelitian yang dilakukan (Fernando & Lukas, 2017) membandingkan algoritma Data Encryption Standard Lightweight (DESL) dan Data Encryption Standard (DES) dengan enkripsi kunci menggunakan RSA, dimana didapatkan hasil bahwa perbedaan waktu komputasi tidak terlalu signifikan. Penelitian (Kurniawan et al., 2018) mencoba mengimplementasikan algoritma baru untuk melakukan enkripsi dan dekripsi. Hasil yang didapatkan menunjukkan proses dekripsi membutuhkan waktu yang lebih lama karena adanya keterbatasan kemampuan komputasi pada perangkat IoT. Penelitian yang mencoba megimplementasikan kriptografi pada sistem IoT juga dilakukan oleh (Endrayanto et al., 2019) yang menguji penggunaan algoritma AES-128 dengan hasil, waktu eksekusi yang relatif lebih cepat dan penggunaan memori yang relatif kecil. Penelitian ini menggunakan modul IoT *Particle Photon*.

Berdasarkan hasil penelitian-penelitian yang sudah dipaparkan sebelumnya, telah diimplementasikan algoritma-algoritma enkripsi pada sistem IoT, seperti algoritma RSA pada perangkat Arduino Duo ((Asang & Sembiring, 2017), algoritma Lizard pada perangkat NodeMCU (Joyoputro et al., 2018), algoritma AES-128 pada modul Particle Photon (Endrayanto et al., 2019), dan algoritma Acorn pada perangkat Wemos ESP8266 (Hananto et al., 2019). Pada penelitian ini, diimplementasikan algoritma AES-128 pada mikrokontroler NodeMCU.

Tujuan penelitian ini adalah mengamankan suatu sistem IoT dengan mengimplementasikan algoritma AES-128 melalui studi kasus pengiriman data pertanian yaitu data suhu dan kelembaban. Ini termasuk data yang dapat dimanfaatkan pada implementasi *Smart Farm* selain curah hujan dan intensitas cahaya (Wedashwara et al.,

2019). Sedangkan pemilihan algoritma AES-128 didasarkan pada kelebihan yang dimiliki yaitu lebih cepat daripada algoritma lainnya seperti DES (Pammu et al., 2017), Blowfish, Two Fish, dan Serpent (Roy et al., 2014). Adapun untuk perangkat yang digunakan yaitu mikrokontroler NodeMCU dengan arsitektur komunikasi menggunakan *web service* REST API. NodeMCU merupakan mikrokontroler berbasis ESP 8266 dilengkapi dengan USB *port* untuk memasukkan program dan *power supply* (Guna et al., 2018). NodeMCU membetuk satu paket board yang kompak dengan berbagai fitur layaknya mikrokontroler (Marina et al., 2020). Sementara itu, REST API merupakan arsitektur *web service* dengan hubungan *client* dan *server*. *Client* akan meminta *request* ke *server* kemudian akan diproses dan direspon oleh *server* (Wardhana et al., 2020). Untuk melihat performa algoritma AES 128 ini, dilakukan pengujian fungsionalitas, pengujian waktu komputasi dan memori. Selain itu dilakukan pengujian keamanan pada sistem IoT yang telah dibuat dengan skenario *sniffing*.

METODE

Penelitian ini menggunakan metode eksperimental dengan mensimulasikan sistem IoT yang dapat membaca data pertanian yaitu suhu dan kelembaban tanah. Metode ini dapat menguji secara benar, hipotesis menyangkut hubungan kasual (Givy Devira Ramady et al., 2020). Tahapan-tahapan pada metode penelitian ini yaitu, pada tahap pertama, dilakukan studi literatur untuk mengumpulkan berbagai referensi yang sesuai dengan penelitian ini. Kedua, analisis kebutuhan sistem yang terdiri dari analisis kebutuhan perangkat keras dan perangkat lunak dalam pengembangan lingkungan komunikasi pada *web service* REST. Ketiga, dilakukan perancangan sistem meliputi perancangan perangkat keras, perangkat lunak, dan perancangan *database*. Keempat, dilakukan implementasi dari perancangan yang dilakukan pada sisi *client* dan *server* REST. Setelah itu, akan dicek apakah fungsi sudah berjalan dengan baik, jika ya, maka dilanjutkan ke langkah selanjutnya, jika tidak, maka dilakukan perbaikan pada perancangan. Kelima, pengujian sistem dengan beberapa pengujian meliputi fungsionalitas, penghitungan waktu komputasi dan memori serta pengujian sisi keamanan. Terakhir, dilakukan penyelesaian dokumentasi penelitian.

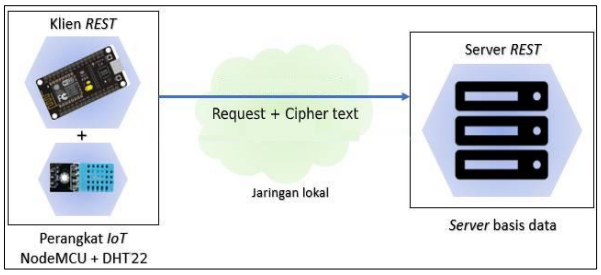
Analisis Kebutuhan Sistem

Pada penelitian ini dibutuhkan perangkat keras untuk pengimplementasian algoritma AES 128-bit dan pembangunan sistem IoT. Kebutuhan perangkat keras pada penelitian ini yaitu, sensor DHT-22 untuk membaca suhu dan kelembaban; NodeMCU 1.0 sebagai mikrokontroler; laptop sebagai server REST dan basis data; serta *router* wifi.

Kebutuhan perangkat lunak untuk pengembangan sistem IoT dan arsitektur komunikasi REST yaitu Linux Ubuntu sebagai sistem operasi; IDE Arduino sebagai lingkungan pengembangan untuk pembuatan program dan implementasi algoritma AES 128 bit; Visual Studio Code sebagai editor untuk menuliskan kode-kode perintah; Microframework Flask untuk pengembangan lingkungan *web service* REST; dan Wireshark sebagai program untuk pengujian keamanan dengan proses *sniffing* paket data yang dikirimkan.

Perancangan Sistem

Untuk memudahkan memahami proses komunikasi pada lingkungan sistem yang dibuat, maka dibuat gambaran keterkaitan antar perangkat yang dapat dilihat pada Gambar 1. Pada gambar 1, terdapat hubungan komunikasi antara *client* dan *server* pada lingkungan komuniasi *web service* REST. Sistem berkomunikasi melalui jaringan dimana *Client* akan mengirimkan *request* dengan perintah POST dilengkapi dengan data suhu dan kelembaban yang telah terenkripsi dengan algoritma AES 128 bit menuju *server* REST. *Request* dari *client* akan diterima oleh *server* beserta data dari *client*, kemudian diubah kembali melalui proses dekripsi dan disimpan ke *server* basis data.



Gambar 1. Rancangan arsitektur komunikasi *web service* REST.

Alur Kerja *Server* REST

Beberapa tahapan pada *server* REST yaitu, pertama menerima *request* dan data dalam bentuk *chiper text* dari *client*. Kedua, mendekripsi data (*cipher text*) dengan algoritma AES 128 bit menjadi bentuk *plain text* seperti data semula. Ketiga, menyimpan data kelembaban dan suhu ke dalam *database*. Jika sistem atau *server* berhenti, maka proses pada keseluruhan sistem akan terhenti.

Alur Kerja *Client* REST

Tahapan-tahapan pada client REST yaitu, pertama membuat koneksi ke jaringan Wifi, Kedua, membaca nilai kelembabab dan suhu yang didapatkan oleh sensor DHT-22. Ketiga, mengenkripsi data suhu dan kelembaban menggunakan algoritma AES 128 bit sehingga bentuknya berubah menjadi *chiper text*. Keempat, mengirimkan *request* dan *resource* dalam bentuk *chiper text* menuju *server* dengan metode POST. Jika sistem dihentikan ,maka proses akan terhenti.

HASIL DAN PEMBAHASAN

Hasil

Terdapat 3 komponen utama perangkat keras pembentuk sistem. Pada sisi *client* yaitu modul sensor DHT-22 dan mikrokontroler NodeMCU, sedangkan pada sisi *server* yaitu sebuah laptop yang bertindak sebagai *web server* REST. Pada sisi *client* dapat dilihat sebuah modul sensor DHT-22 yang dihubungkan ke mikrokontroler NodeMCU melalui pin-pin yang ada pada kedua perangkat menggunakan kabel *jumper*.

Realisasi Perancangan Perangkat Keras

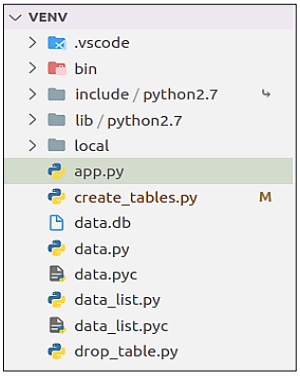
Hasil analisis kebutuhan perangkat keras, maka dapat direalisasikan dalam bentuk rangkaian yang dapat dilihat pada gambar 2. Gambar 2 menunjukkan penyusunan perangkat keras pada sisi *client* dan *server*. Terdapat 3 komponen utama yaitu, sisi *client* berupa modul sensor DHT-22 dan mikrokontroler NodeMCU, sedangkan pada sisi *server* yaitu sebuah laptop yang bertindak sebagai WEB server REST.

Gambar 2. Realisasi Penyusunan Perangkat Keras Sistem *Client* dan *Server*

Pada sisi *client* dapat dilihat sebuah modul sensor DHT-22 yang dihubungkan ke mikrokontroler NodeMCU dengan menggunakan pin yang ada pada perangkat menggunakan kabel *jumper*. Pin 3V pada NodeMCU terhubung ke pin positif pada DHT-22. Sedangkan pin GND pada NodeMCU terhubung dengan pin negative pada DHT-22. Pin D4 pada NodeMCU terhubung dengan pin *out* pada DHT-22. NodeMCU mendapatkan daya listrik melalui kabel USB yang terhubung dengan laptop dan sekaligus digunakan untuk memasukkan kode program ke NodeMCU.

Realisasi Perancangan Perangkat Lunak Server

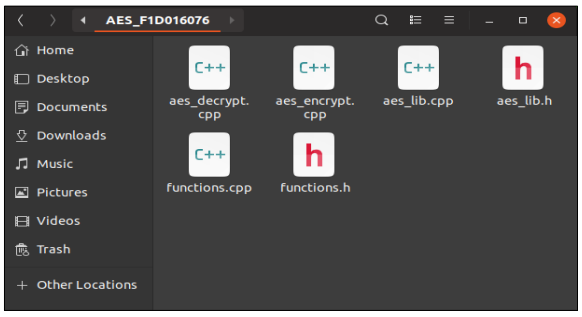
Pada realisasi perancangan perangkat lunak sisi server, digunakan micro-framework Flask sebagai kerangka kerja untuk membangun *web service* REST. Struktur kerangka kerja dari Flask dapat dilihat pada gambar 3.



Gambar 3. Struktur kerangka kerja Flask

Realisasi Algoritma AES 128

Untuk dapat menggunakan algoritma AES 128 bit, dibuat sebuah *library* algoritma AES 128-bit menggunakan bahasa C pada Arduino IDE sesuai dengan perancangan sebelumnya. *Library* ini berfungsi untuk melakukan proses enkripsi data pada *client*. Hasil realisasi perancangan algoritma AES 128-bit dapat dilihat pada gambar 4. Setelah *library* algoritma AES 128 dibuat, selanjutnya *library* ditempatkan pada lokasi *library* Arduino IDE agar dapat terbaca dan digunakan untuk melakukan proses enkripsi pada *client*. Berikut diperlihatkan proses include library yang sudah dibuat pada arduino IDE pada Gambar 5.

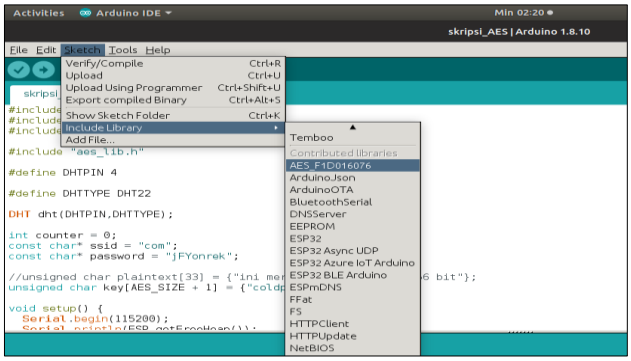


Gambar 4. Hasil realisasi perancangan algoritme AES 128 bit

Pengujian Waktu

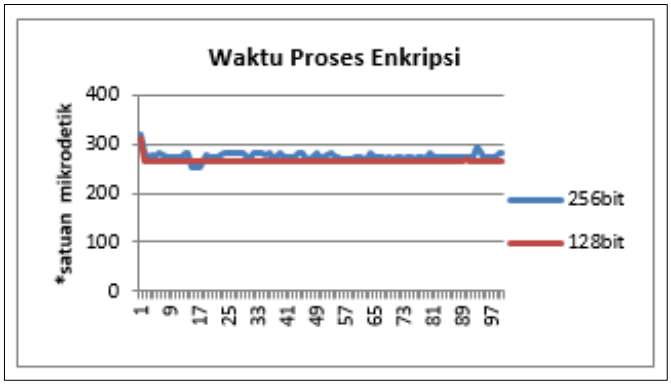
Pada skenario pengujian kinerja waktu dilakukan dengan cara mengambil waktu proses enkripsi sebanyak 100 kali pengujian, kemudian hasilnya dirata-rata. Tujuan dilakukan lebih dari satu kali yaitu untuk memperoleh sampel kinerja waktu yang presisi dan stabil. Proses pengambilan waktu ini dilakukan dengan cara mengurangi waktu setelah dilakukan enkripsi dengan waktu sebelum dilakukan enkripsi, sehingga didapatkan waktu yang digunakan dalam

proses enkripsi. Karena AES menetapkan *block input* harus 128 bit maka ukuran *plain text* yang digunakan dalam pengujian ini sebesar 128 dan 256 bit. Ukuran *plain text* ini digunakan sebagai perbandingan kinerja waktu proses enkripsi.



Gambar 5. Proses *Include Library* AES 128 bit

Gambar 6 menggambarkan hasil pencatatan waktu eksekusi saat proses enkripsi *plain text* sebesar 128 dan 256 bit. Waktu komputasi enkripsi *plain text* dengan ukuran 128 bit yaitu 266.31 mikrodetik. Sedangkan waktu komputasi enkripsi *plain text* dengan ukuran 256 bit yaitu 274.31 mikrodetik.



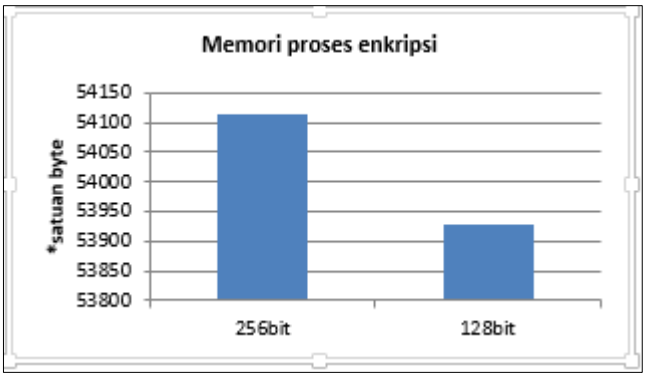
Gambar 6. Grafik Hasil Pengujian Kinerja Waktu Proses Enkripsi

Pengujian Kinerja Memori

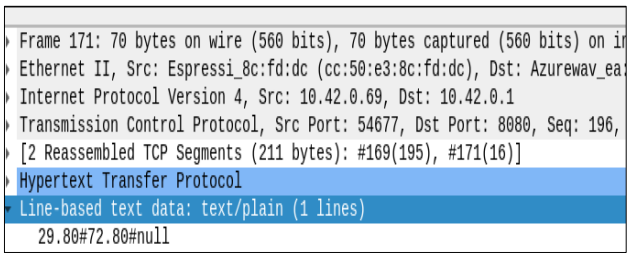
Skenario pengujian dilakukan dengan cara mengambil jumlah memori yang digunakan dalam proses enkripsi sebanyak 100 kali pengujian, kemudian dirata-rata. Pengujian dilakukan lebih dari satu kali bertujuan untuk memperoleh sampel kinerja memori yang presisi dan stabil. Proses pengambilan jumlah memori dilakukan dengan cara memanggil fungsi ESP.getFreeHeap(). Gambar 7 menggambarkan hasil pencatatan memori yang digunakan pada proses enkripsi *plain text* dengan ukuran 128 dan 256 bit. *Plain text* dengan ukuran 128 bit dienkripsi dengan konsumsi memori sejumlah 53928 *byte*. Sedangkan *plain text* sebesar 256 bit dienkripsi dengan konsumsi memori sebanyak 54114 *byte*.

Pengujian Keamanan

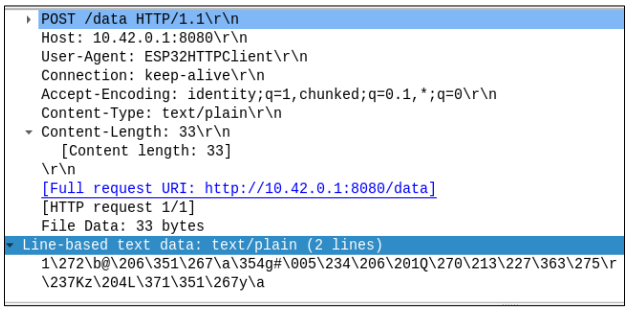
Pada skenario pengujian keamanan dilakukan pengujian dengan metode *sniffing* paket data yang dikirim oleh *client* menuju *server* menggunakan *tool* Wireshark. Dalam hal ini data yang dikirim oleh *client* berupa data suhu dan kelembaban. Paket data yang lewat melalui protokol HTTP kemudian di-*sniffing* dan dilakukan analisis. Berikut diperlihatkan hasil pengujian keamanan untuk melihat bagaimana data tetap dalam bentuk yang tidak dapat dibaca oleh orang yang tidak berhak. Hasil tangkapan layar paket data yang dikirim melalui jaringan wifi dapat dilihat pada gambar 8 dan gambar 9.



Gambar 7. Hasil Pengujian Kinerja Memori.



Gambar 8. Hasil *Capture* Pengiriman Paket Sebelum Enkripsi



Gambar 9. Hasil *Capture* Pengiriman Paket Setelah Enkripsi

Pembahasan

Hasil pengujian yang telah dilakukan pada sistem IoT menunjukkan bahwa pengamanan pada sistem IoT dengan implementasi algoritma enkripsi dapat untuk meningkatkan keamanan data yang ditransmisikan melalui jaringan, hal ini relevan dengan penelitian lain yang telah mengimplementasikan algorima enkripsi lainnya seperti pada (Fernando & Lukas, 2017; Kurniawan et al., 2018; Ravida & Santoso, 2020). Dengan karakteristik yang berbeda dengan lingkungan komputasi *desktop* berupa keterbatasan sumber daya, tidak menghalangi untuk dilakukan mekanisme pengamanan. Sistem IoT tetap dapat berjalan seperti biasa tanpa ada beban yang besar pada proses komputasi.

Adapun untuk pemrosesan, semakin besar ukuran *plain text* yang dienkripsi maka semakin besar pula waktu dan memori yang dibutuhkan, hasil ini relevan dengan penelitian yang dilakukan oleh (Joyoputro et al., 2018). Pada saat menggunakan *plain text* dengan ukuran 128 bit, didapatkan waktu pemrosesan sebesar 266.31 mikrodetik. Sedangkan ketika dilakukan penambahan ukuran *plain text* menjadi 256 bit, didapatkan waktu pemrosesan menjadi 274.31 mikrodetik. Namun, perbedaan waktu tidak terlalu signifikan dengan penambahan ukuran *plain text* 2 (dua) kali lipat. Begitu juga konsumsi memori pada saat pemrosesan dilakukan, pada *plain text* 128 bit, memori yang digunakan yaitu 53928 *byte*. Sedangkan pada pemrosesan *plain text* 256 bit, memori yang digunakan yaitu 54114 *byte*.

Perbedaan penggunaan memori tidak terlalu signifikan walaupun ukuran *plain text* menjadi 2 (dua) kali lipat.

Mekanisme enkripsi dengan menggunakan AES pada sistem IoT dapat memberikan jaminan kerahasiaan data pada mekanisme komunikasi *client-server* berbasis *web service*, hal ini relevan dengan keamanan yang juga diberikan pada mekanisme *publish/subscribe* seperti pada penelitian (Pramukantoro et al., 2019). Bentuk data yang transmisikan pada sistem IoT tidak dapat dibaca setelah dianalisis menggunakan tool *Wireshark*. Dengan algoritma AES 128-bit yang bersifat simetris, data tidak dengan mudah dapat dibaca dan diretas, kecuali jika kunci untuk proses dekripsi diketahui.

SIMPULAN

Kriptografi dapat diimplementasikan pada sistem IoT untuk mengamankan data, pada penelitian ini digunakan algoritma AES 128 bit. Proses enkripsi *plain text* dengan ukuran 128 dan 256 bit membutuhkan waktu 266.31 dan 274.31 mikrodetik, sedangkan memori yang terpakai sebesar 16 % dan 17% dari memori total. Pengimplementasian algoritma AES 128 bit dapat memberikan keamanan yang baik pada data dengan mememuhi unsur *confidentiality* dengan waktu eksekusi yang relatif cepat dan konsumsi memori yang relatif sedikit.

REFERENSI

Asang, M. S., & Sembiring, I. (2017). Keamanan Data Pada Perangkat Internet Of Things Menggunakan Metode Public-Key Cryptography. *Jurnal Teknologi Informasi-Aiti*, *14*(1), 80–87.

Babar, M., & Sohail Khan, M. (2021). ScalEdge: A framework for scalable edge computing in Internet of things–based smart systems. *International Journal of Distributed Sensor Networks*, *17*(7), 1–11. https://doi.org/10.1177/15501477211035332

Bhandari, R., & Kirubanand, V. B. (2019). Enhanced encryption technique for secure iot data transmission. *International Journal of Electrical and Computer Engineering*, *9*(5), 3732–3738. https://doi.org/10.11591/ijece.v9i5.pp3732-3738

Endrayanto, R. K., Muttaqin, A., & Setyawan, R. A. (2019). Advanced Encryption Standard (AES) pada Modul Internet of Things (IoT). *TELKA - Telekomunikasi, Elektronika, Komputasi Dan Kontrol*, *5*(2), 103–113. https://doi.org/10.15575/telka.v5n2.103-113

Fernando, & Lukas. (2017). Implementasi dan Analisis Lightweight Cryptography untuk Internet of Things (IOT). *Jurnal Elektro*, *10*(2), 85–94.

Givy Devira Ramady, Andrew Gea Mahardika, Lestari, N. S., & Syafrudin. (2020). Perancangan Model Simulasi Sistem Pengendali Suhu Ruang Kelas Berbasis Internet Of Things. *Seminar Nasional Riset Teknologi Terapan*, *1*(1).

Guna, P. I. A., Suyadnya, I. M. A., & Agung, I. G. A. P. R. (2018). Sistem Monitoring Penetasan Telur Penyu Menggunakan Mikrokontroler NodeMCU ESP8266 dan Protokol MQTT dengan Notifikasi Berbasis Telegram Messenger. *Journal of Computer Science and Informatics Engineering (J-Cosine)*, *2*(2), 80–89. https://doi.org/10.29303/jcosine.v2i2.135

Hananto, M. A., Kusyanti, A., & Primananda, R. (2019). Implementasi Algoritme Acorn untuk Pengamanan Data pada Protokol MQTT menggunakan Perangkat Wemos ESP8266. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *3*(8), 7869–7878.

Joyoputro, K., Kusyanti, A., & Bakhtiar, F. A. (2018). Implementasi Algoritme Kriptografi Lizard untuk Mengamankan Pengiriman Data Menggunakan Arsitektur Web Service REST pada Mikrokontroler NodeMCU. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *2*(12), 6292–6299.

Kurniawan, A., Mayasari, R., & Murti, M. A. (2018). Implementation of Cryptographic Algorithm on IoT Device's ID. *Jurnal Sistem Cerdas*, *1*(2), 20–28.

Marina, A., Ilman, H. K., Febi, F., Muhammad, A. E., & Muhammad, I. (2020). Studi Perbandingan Platform Internet of Things (IoT) untuk Smart Home Kontrol Lampu Menggunakan NodeMCU dengan Aplikasi Web Thingspeak dan Blynk. *Jurnal Fidelitiy*, *2*(1), 59–78.

Meutia, E. D. (2015). Interet of Things – Keamanan dan Privasi. *Seminar Nasional Dan Expo Teknik Elektro*, 85–89.

Pammu, A. A., Chong, K. S., Ho, W. G., & Gwee, B. H. (2017). Interceptive side channel attack on AES-128 wireless communications for IoT applications. *IEEE Asia Pacific Conference on Circuits and Systems, APCCAS 2016*, 650–653. https://doi.org/10.1109/APCCAS.2016.7804081

Pramukantoro, E. S., Bakhtiar, F. A., Aji, A. L. B., & Dewa, D. H. P. (2019). Implementasi Mekanisme End-To-End Security pada IoT Middleware. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *6*(3), 335–340. https://doi.org/10.25126/jtiik.2019631401

Ravida, R., & Santoso, H. A. (2020). Advanced Encryption Standard (AES) 128 Bit for Hydroponic Plant Internet of Things (IoT) Data Security. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *4*(6), 1157–1164.

Roman, R., Najera, P., & Lopez, J. (2017). Securing the internet of things. *Smart Cards, Tokens, Security and Applications: Second Edition*, *September*, 445–468. https://doi.org/10.1007/978-3-319-50500-8_16

Roman, R., Zhou, J., & Lopez, J. (2013). On the features and challenges of security and privacy in distributed internet of things. *Computer Networks*, *57*(10), 2266–2279. https://doi.org/10.1016/j.comnet.2012.12.018

Roy, D., Paul, S., & Das, S. (2014). A comparative study of AES , Blowfish , Two fish and serpent cryptography algorithms. *Elixir IInformation Technology*, *72*, 25218–25219.

Skirelis, J., & Navakauskas, D. (2017). Edge computing in IoT: Preliminary results on modeling and performance analysis. *Proceedings of the 5th IEEE Workshop on Advances in Information, Electronic and Electrical Engineering*, 1–4. Riga, Latvia: IEEE. https://doi.org/10.1109/AIEEE.2017.8270555

Wahyudi, F., & Utomo, L. T. (2021). Perancangan Security Network Intrusion Prevention System Pada PDTI Universitas Islam Raden Rahmat Malang Farid. *Edumatic : Jurnal Pendidikan Informatika*, *5*(1), 60–69. https://doi.org/10.29408/edumatic.v5i1.3278

Wardhana, W. G., Arwani, I., & Rahayudi, B. (2020). Implementasi Teknologi Restful Web Service Dalam Pengembangan Sistem Informasi Perekaman Prestasi Mahasiswa Berbasis Website (Studi Kasus : Fakultas Teknologi Pertanian Universitas Brawijaya). *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *4*(2), 680–689.

Wedashwara, W., Ahmadi, C., & Arimbawa, I. W. A. (2019). Sequential fuzzy association rule mining algorithm for plants environment classification using internet of things. *AIP Conference Proceedings*, 1–10. https://doi.org/10.1063/1.5141287

Wei, H., Luo, H., Sun, Y., & Obaidat, M. S. (2020). Cache-Aware Computation Offloading in IoT Systems. *IEEE Systems Journal*, *14*(1), 61–72. https://doi.org/10.1109/JSYST.2019.2903293